

IS 2900 – Project on IT Applications

Final Report

School Management System

Group Kripto

Index No	Name
175053T	Nasrina M.I.F
175064D	Prasandika K.A.C
175083J	Thilakarathna W.M.D.S
175090D	Welagedara J.M

Supervised by:

Mr. S.M.U. Premasiri
Ms. W.A.S.N. Wijethunge

Client:

Arthur C Clark Institute for Modern Technologies
Katubedda, Moratuwa

Faculty of Information Technology
University of Moratuwa
2022

Declaration

We declare that this report is our own work and has not been submitted in any form for another degree or diploma at any university or other institution of tertiary education. Information derived from the published or unpublished work of others has been acknowledged in the text and a list of references is given.

Names of Students

175053T - Nasrina M.I.F

Signatures of Students



175064D - Prasandika K.A.C



175083J - Thilakarathna W.M.D.S.



175090D – Welagedara J.M



Date: 18/06/2022

Supervised by:

Names of Supervisors

S.M.U.Premasiri

Date:

Signatures of Supervisors

W.A.S.N. Wijethunge

Date:

Abstract

In the current world, parents do not have enough time to pay attention to their children's school activities and studies which is very important for their future. Though there are learning management systems like My class campus, fedena, Moodle etc. to support this issue, there are several disadvantages in them basically because those are based on monolithic architecture. Therefore, through the proposed system a web-based school management system will be developed to manage the essential activities in a school learning system more efficiently using MVC architecture. Basically students, parents, teachers and administration who are the users of the system, are able to observe the progress of students. EJS, HTML, Java Script, CSS and Node JS are used in developing the frontend web and backend of the application respectively. Therefore, through the proposed system the drawbacks in the current school management systems will be avoided while establishing a strong relationship among students, parents, teachers and school administration.

Table of Contents

Chapter 1 Introduction.....	8
1.1 Introduction	8
1.2 Problem in Brief	9
1.3 Aim and Objectives	9
1.3.1. Aim	9
1.3.2. Objectives.....	9
1.4 Proposed Solution	10-11
1.5 Resource requirements	11
1.6 Summary	12
Chapter 2 Literature Review	13
2.1 Introduction	13
2.2 Existing Systems	13
2.2.1 MyClassCampus	13
2.2.2 Fedena	13
2.2.3 School Tronic.....;	14
2.2.3. Moodle	14
2.3 Summary	14
Chapter 3 Our Approach	15
3.1 Introduction	15
3.2 Users	15
4.3 Inputs	15
3.4 outputs	15
3.5 process	15
3.6 Technology	16
3.6.1 EJS.....	16
3.6.2. JavaScript	16
3.6.3. HTML & CSS	16
3.6.4. Node JS.....	16
3.7 Summary	17
Chapter 4 Analysis and Design.....	18
4.1 Introduction	18
4.2 Analysis	18-19
5.3 Design	19
5.3.1 ER diagram.....	20
5.3.2 System diagram	21
5.3.3 Use case diagram	22
5.3.4 Activity diagrams	23-36
5.3.5 Class diagram	37

5.3.6 Sequence diagram	38-48
5.4 Summary	48
Chapter 5 Implementation	49
5.1 Introduction	49
5.1.1 Interfaces.....	50-64
5.1.2 Code Segments.....	65-73
6. 2 Summary	73
Chapter 6 Discussion and Conclusions	74
References	75
Appendix	

List of Figures/Tables

Figure1:Proposed Solution.....	11
Figure 2:Fedena school administration software.....	14
Figure3:Moodle.....	15
Figure 4:ER Diagram.....	21
Figure 5:System diagram.....	22
Figure 6:Use case diagram.....	23
Figure 7:Activity diagram for all users.....	24
Figure 8:Activity diagram for parent to do payment.....	25
Figure 9:Activity diagram for parent to view progress reports of students.....	26
Figure 10:Activity diagram for Student to view homework & assignments.....	27
Figure 11:Activity diagram for Student to view notices.....	28
Figure 12:Activity diagram for Student to post creative activities.....	29
Figure 13: Activity diagram for Student to access to quizzes.....	30
Figure 14: Activity diagram for Teacher to upload notices.....	31
Figure 15: Activity diagram for Teacher to upload assignments/homework/quizzes.....	32
Figure 16: Activity diagram for Teacher to view submission of students.....	33
Figure 17: Activity diagram for Teacher to mark attendance.....	34
Figure 18: Activity diagram for Admin to generate and view reports.....	35
Figure 19: Activity diagram for Admin to handle payments.....	36
Figure 20: Activity diagram for Admin to handle user registration.....	37
Figure 21: Class diagram.....	38
Figure 22: Login and reset password.....	39
Figure 23: Edit user profile.....	40
Figure 24: View reports.....	41
Figure 25: View articles and notices.....	42
Figure 26: Register new user.....	43
Figure 27: Assign classes and subjects.....	44
Figure 28: Upload activities.....	45
Figure 29: View assignment and upload answers.....	46
Figure 30: Upload articles.....	47
Figure 31: Mark attendance.....	48
Figure 32: Upload learning materials.....	49
Figure 33: Upload news.....	49
Figure 34: UI for user login	48
Figure 35: UI for Admin login	49
Figure 36: UI for Admin registration.....	49
Figure 37: UI for Forgot password.....	50

Figure 38: UI for Admin overview.....	51
Figure 39: UI for Admin profile.....	52
Figure 40: UI for manage profiles.....	52
Figure 41: UI for manage courses.....	53
Figure 42: UI for manage classes.....	53
Figure 43: UI for student login.....	54
Figure 44: UI for student forgot password.....	54
Figure 45: UI for student reset password.....	55
Figure 46: UI for student overview.....	55
Figure 46: UI for student profile.....	56
Figure 47: UI for student upload assignments and homework.....	56
Figure 48: UI for student view assignments and homework.....	57
Figure 49: UI for student view and download assignments and homework.....	57
Figure 50: UI for parent login.....	58
Figure 51: UI for parent forgot password.....	58
Figure 52: UI for parent password resetting link.....	59
Figure 53: UI for student reset password.....	59
Figure 54: UI for student reset password.....	60
Figure 55: UI for teacher login.....	60
Figure 56: UI for teacher forgot password.....	61
Figure 57: UI for teacher reset password.....	61
Figure 58: UI for teacher overview.....	62
Figure 59: UI for teacher profile.....	62
Figure 60: UI for teacher mark attendance.....	63
Figure 61: UI for teacher view submitted assignments.....	63
Figure 62: UI for teacher upload assignments and homework.....	63
Figure 63: Code segment for the database1.....	64
Figure 64: Code segment for the database2.....	65
Figure 65: Code segment for the database3.....	66
Figure 66: Code segment controllers - Parent.....	67
Figure 67: Code segment controllers - Staff.....	68
Figure 68: Code segment controllers - Student.....	69
Figure 69: Code segment Authentication - Admin.....	70
Figure 70: Code segment Authentication - Parent.....	70
Figure 71: Code segment Authentication - Staff.....	71
Figure 72: Code segment Authentication - Student.....	72

Chapter 1

Introduction

1.1 Introduction

The education of children is a significant factor in the life of them. But in the modern world individuals in each and every society are spending a very busy life style. As a result, parents do not have enough time to pay attention to children's progress in studies. Therefore, in order to strengthen the relationship between the parents, students, teachers, school administration and to facilitate paperless administration of schools, a school management system is proposed as a solution. There are many learning management systems which have already been developed. But in those systems parents are not included as a user. Therefore, in the proposed school management system parents, students, teachers and administration are allowed to access the system. The proposed system will handle only the administration of the educational activities of students. The aim of the system is to develop a web-based system to manage the essential activities in a school learning system more efficiently and effectively. The objectives of developing this system are, developing a system which can be used to observe the progress of a student, teachers and parents, automate the administrative activities which are currently done manually. The rest of the paper is organized as follows. Section 2 reviews the related literature. Section 3 describes the technology adapted. Section 4 discusses the approach. Section 5 includes analysis and design. Section 6 is about the implementation. Section 7 is on evaluation of the system.

1.2 Problem in Brief

Nowadays school progress meetings are getting quarterly. Therefore, we can see the gap between teachers and parents. Then it has limited time to discuss the progress of the student with the parent. As well as parents cannot get continuous progress about the student when it happens quarterly. Within this time there can be lot of changes of that particular student and sometimes there can miss or do not remind the special things of student, since there is a huge gap between two quarters. Therefore, parents cannot get a true picture about students continuously. Students may leave home to school. But he has not attended to the class and does not do activities within the class. Then that also a big problem. As usual if the student does it, that is badly affected to the student. Parents could not view the daily attendance of the student. When it is dealing with manual progress reports, sometimes students change their results if they have got low marks. That is a major problem today.

Most of the time parents cannot see the students' creative works except exhibition when they doing in school times. So, it is better to have displayed these works through a dashboard. Then parents can see students' works and motivate them more and more to do creative activities. Going to implement a School Management System by addressing above mentioned problems.

1.3 Aim and Objectives

1.3.1. Aim

Developing a web-based system which can be used to observe the clear progress of a student and automate the administrative activities which are currently done manually.

1.3.2. Objectives

- To develop user registration and login service
- To develop student profile
- To develop dashboard
- To develop report generation service
- To develop user management service

1.4 Proposed Solution

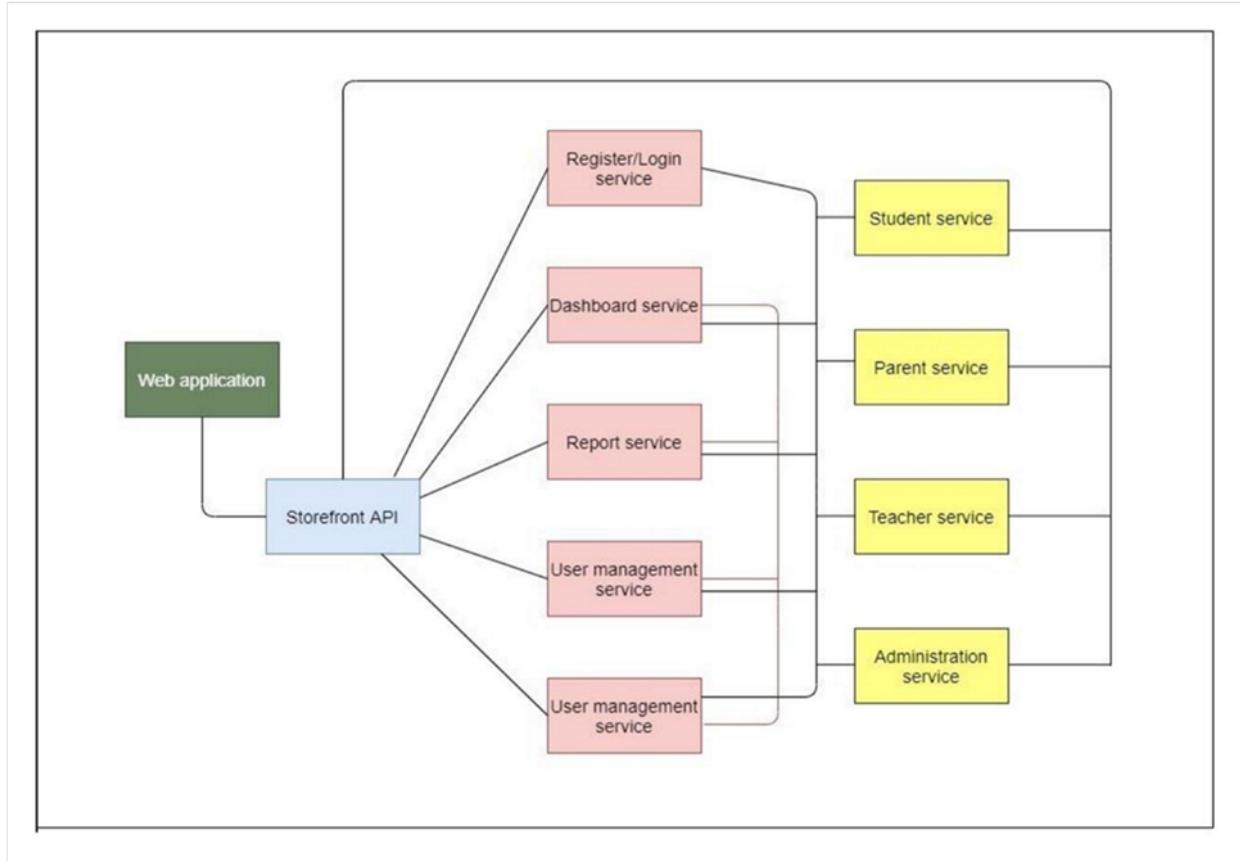


Figure 1: Proposed Solution

Registration Service: - When student, teacher or an administration member is enrolled here we keep records like their admission number, name, family member details and other details are recorded. Parent also get access to this system with the registration of the student.

Report Service: - System will auto generates the reports to analyze student's performance and progress. After generating these reports, it will able to compare the reports with each student and each term. There will be term wise exam result reports and assignment marks reports for each student. They can be viewed only by the relevant student, parent and the teacher. Attendance reports of each student will be generated monthly. Term wise marks of each subject and average marks of each subject of each class can be viewed as charts. These charts can be viewed only by the relevant teacher. So, using these charts teacher can identify the places that should be change for the betterment of the class.

Dashboard and notice board Service: - In a nutshell, this is a notice board. Students can upload their creative works to the dashboard. Admins and teachers can upload special notices in the noticeboard. At the time it is uploaded no one can view what has uploaded to the dashboard. It appears on the dashboard after the article is approved by an admin. Achievements that are acquired also appear on this. This can be seen by every party in this system.

User Management Service: -Manages user login for parents, staff, administrators and students with customizable role privileges. Assigns privileges for predefined roles to carry out relevant operations.

Student service: - Compare their own exam, assignment marks, attendance records, achievements via reports. Students can know about the home work and upcoming events in the school.

Parent service: - Parents can access, the system and view their children's attendance and reports at the same time. Special events in the school can be viewed via dashboard.

Teacher service: - In the teacher service, a teacher will upload assignments, quizzes and exam papers. After marking the papers marks will be uploaded by the teacher. At the same time teacher will mark the attendance of the students from his or her mobile phone.

Administration service: -When a new student, teacher or a parent is registered it will be done by the administrative service. After a teacher upload exam results and students reports, it will be checked and make them visible in the dashboard by the administrative service.

1.5 Resource requirements

- Computer with 4GB RAM, 2.4GHz or more processing power
- Backend: NodeJS
- Frontend: JavaScript, React, EJS, HTML, CSS
- Database: MySQL

1.6 Summary

This chapter provide a brief description about our project. Our main goal is to use microservices architecture to overcome the disadvantages of monolithic architecture.

In chapter 2 it describes about other school management systems.

In chapter 3 it describes about the technology we use.

In chapter 4 it describes about the approach for the problem domain.

Chapter 5 is about the analysis and design of the system.

Chapter 6 is on the implementation of the system

Chapter 7 is on test results of the system

Chapter 2

Literature Review

2.1 Introduction

There are many open-source free and paid school management systems. Each of them performs different tasks in a school.

2.2 Existing Systems

2.2.1 MyClassCampus

MyClassCampus is a leading platform to provide school management. Also providing facilitate to conversion between students, teachers, and parents for incredible correspondence and general productivity. Designed with advanced features to save the time of admin and gives access to vital information on a single click. The thing is there is no specific facilities to school activities.

2.2.2 Fedena

Fedena is an open-source school administration software that largely focuses on handling records.



Figure 2: Fedena school administration software

2.2.3. School Tonic

School Tonic is web-based school management software which designed to facilitate paperless administration of schools. It has the ability to maintain academic history of students, staffs, various records, etc.

2.2.4. Moodle

Moodle is a free and open-source learning management system written in PHP and distributed under the GNU General Public License. This allows for extending and tailoring learning environments using community sourced plugins.



Figure 3: Moodle

2.3 Summary

There are many more school management software which were designed based on the customer specific requirements.

Chapter 3

Our Approach

3.1 Introduction

This project is designed to facilitate paperless administration of schools. School management software has the ability to host modules that allow a user to maintain academic, history of students, records, etc. Each student, teacher, parent and administrators can register to the system comes in mobile application and web interface which is implemented by react native and react.

3.2 Users

This system is used by students, teachers, parents and admins.

4.3 Inputs

User details, notes, examination, marks, marking scheme, attendance records, articles, news, student answers

3.4 outputs

Progress report, attendance report

3.5 process

Students can enroll to the system under the supervision of admin. Teachers, parents and admins are also given a username and password, so they also become users of this system. Teachers upload student activities (marks, notes, examination, announcements, marking schemes, attendance).

Then students are able to see them. Parents are able to view their child's attendance reports and progress reports which are created by the system automatically. Students can upload their assignments then teachers can correct them and release marks. Students can upload their articles. Teachers and admins can upload news. Confirmation for the articles and news is given by the admin and admin is the main role for user registration. All users can view articles and news.

3.6 Technology

School management system consists of a web-based application. For the frontend we used EJS , JavaScript, HTML and CSS. For database we used SQL.

3.6.1 EJS

EJS simply stands for Embedded Javascript. It is a simple templating language/engine that lets its user generate HTML with plain javascript. EJS is mostly useful whenever you have to output HTML with a lot of javascript. EJS is a tool for generating web pages that can include dynamic data and can share templated pieces with other web pages (such as common headers/footers). It is not a front-end framework.

3.6.2. JavaScript

JavaScript, often abbreviated as JS, is a programming language that is one of the core technologies of the World Wide Web, alongside HTML and CSS. Over 97% of websites use JavaScript on the client-side for web page behavior, often incorporating third-party libraries.

3.6.3. HTML and CSS

The HyperText Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets and scripting languages such as JavaScript.

Cascading Style Sheets is a style sheet language used for describing the presentation of a document written in a markup language such as HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

3.6.4. Node JS

Node.js is primarily used for non-blocking, event-driven servers, due to its single-threaded nature. It's used for traditional web sites and back-end API services, but was designed with real-time, push-based architectures in mind. Node.js is an open source server environment; Node.js is free; Node.js runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)

3.7 Summary

In this chapter we described about the input, output, process and technology that we are going to use in our system.

Chapter 4

Analysis and Design

4.1 Introduction

After identifying the problem, we gathered the requirements from our client and started developing a solution. In order to gain a clear idea about the system, we designed our system using UML diagrams. We designed both the structural and behavioral diagrams.

4.2 Analysis

Before designing the system, we cleared out the user requirements that we address from our project.

Functional requirements

- All users should be able to login
- All users should be able to edit profile
- All users should be able to view notices
- All users should be able to view progress reports
- All users should be able to view attendance reports
- All users should be able to view articles
- All users should be able to view timetable
- Student should be able to view homework and assignments
- Student should be able to upload homework and assignments
- Student should be able to view notices
- Student should be able to publish creative activities
- Student should be able to View Assignment marks
- Student should be able to View timetable
- Student should be able to view upcoming events in the school
- Student should be able to access the quiz
- Student should be able to view dashboard.
- Parent should be able to view timetable
- Parent should be able to view marks
- Parent should be able to view dashboard.

- Parent should be able to view creative activities of all students
- Parent should be able to donate
- Parent should be able to pay school fees
- Parent should be able to chat with teacher
- Teacher should be able to mark and upload attendance
- Teacher should be able to upload assignments and homework
- Teacher should be able to upload notices
- Teacher should be able to view students' assignments and homework
- Teacher should be able to upload quizzes
- Teacher should be able to upload marks

Non-functional requirements

- User interfaces should be user-friendly
- Less response times
- Security and privacy

5.3 Design

Using the requirements we gathered, we drew the UML diagrams to indicate our proposed solution.

5.3.1 ER diagram

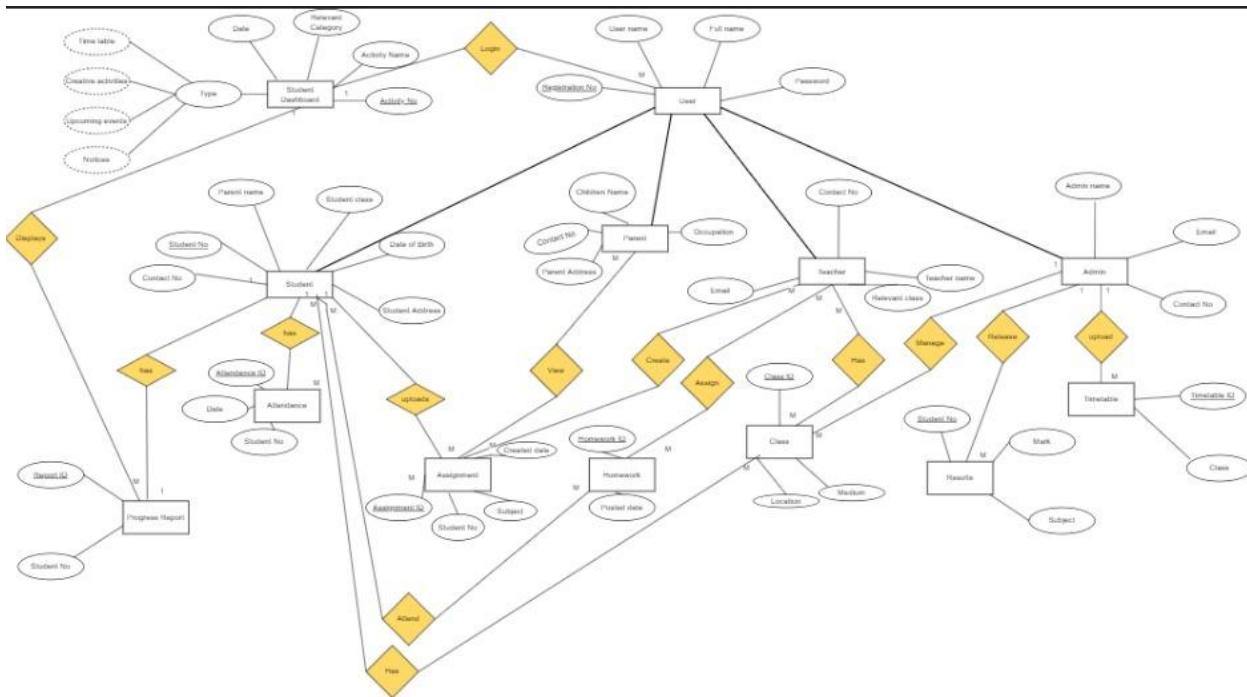


Figure 4: EER Diagram

5.3.2 System diagram

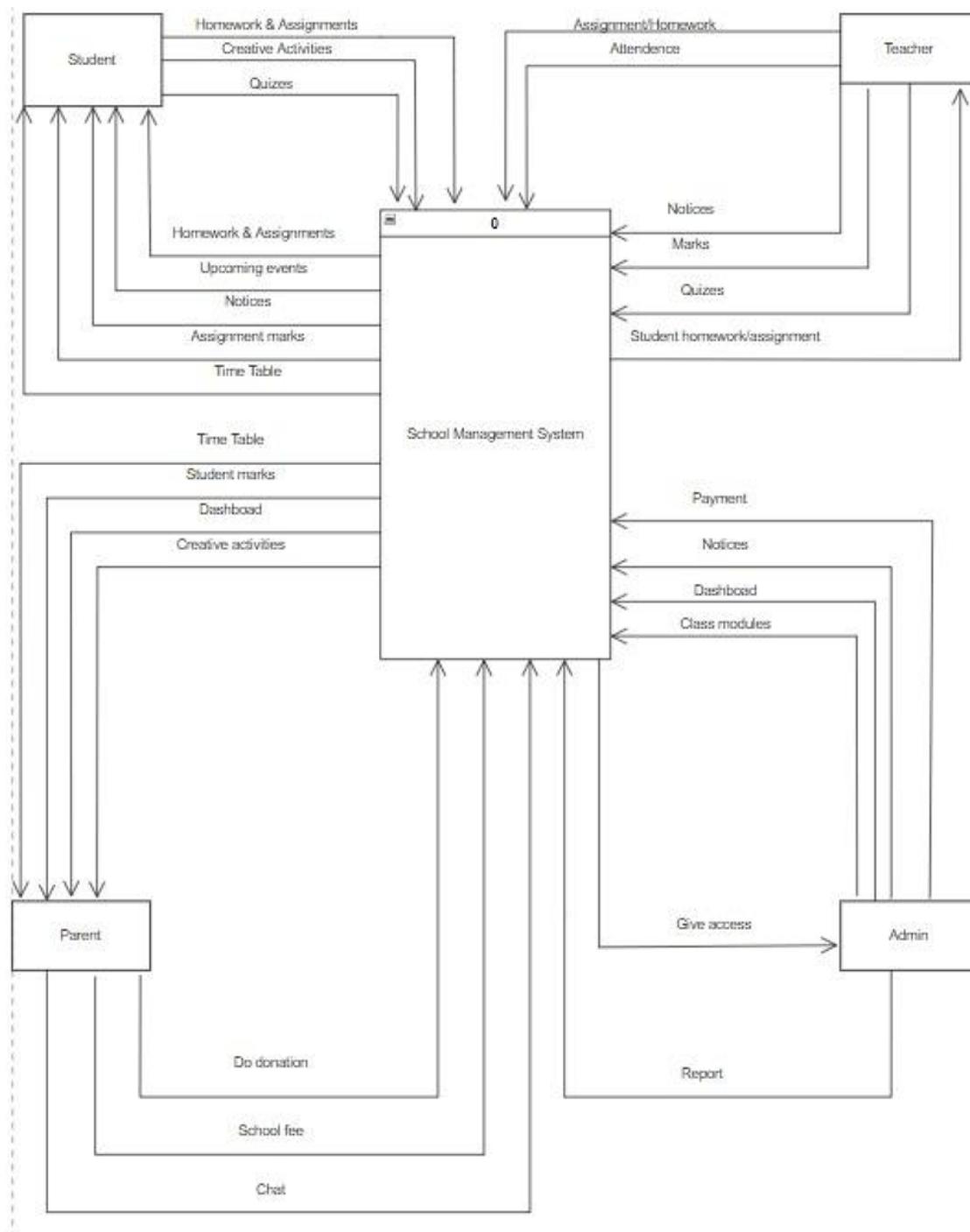


Figure 5: System diagram

5.3.3 Use case diagram

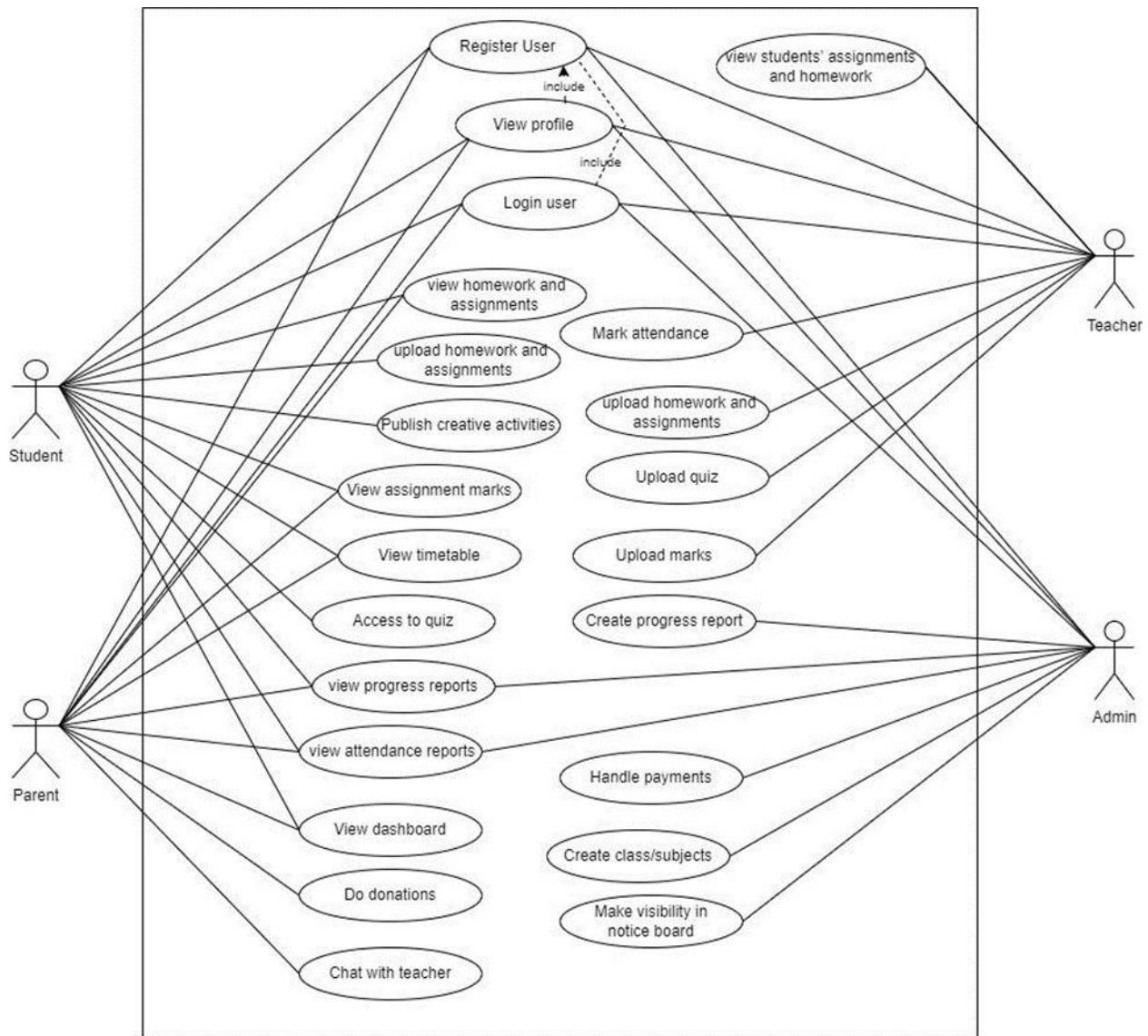


Figure 6: Use case diagram

5.3.4 Activity diagrams

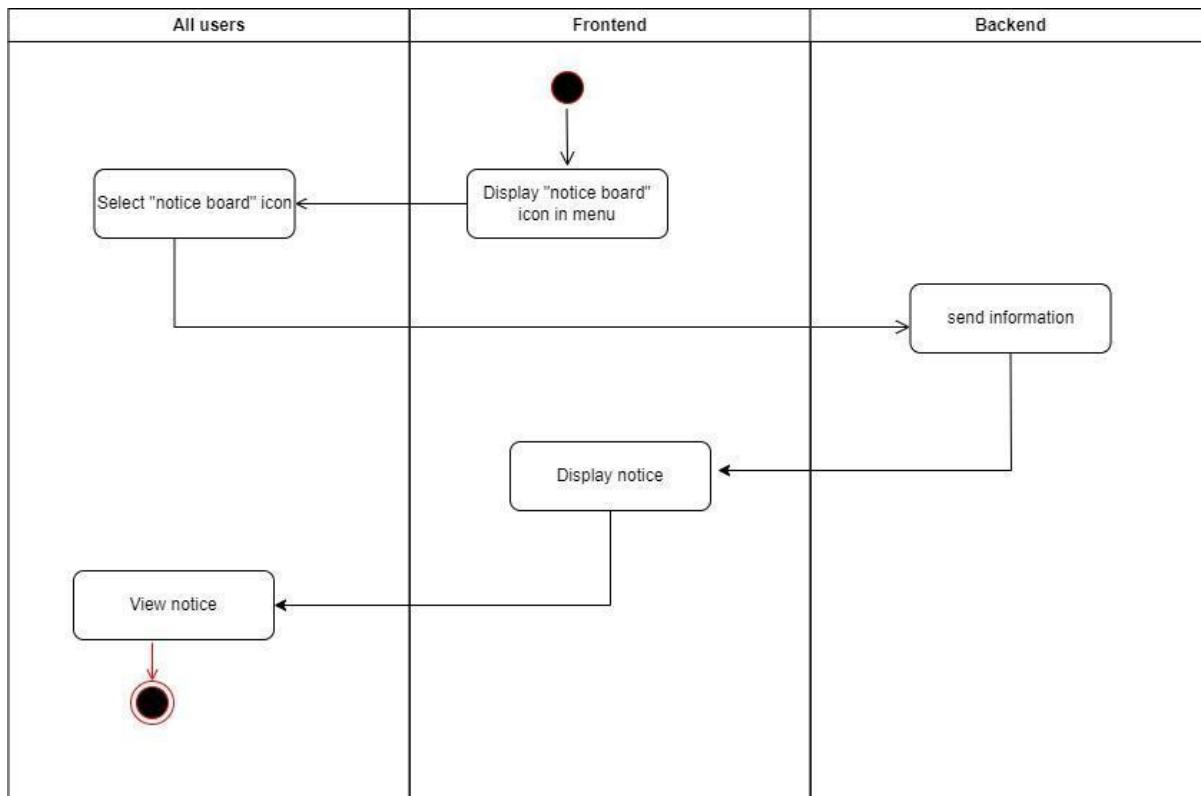


Figure 7: Activity diagram for all users

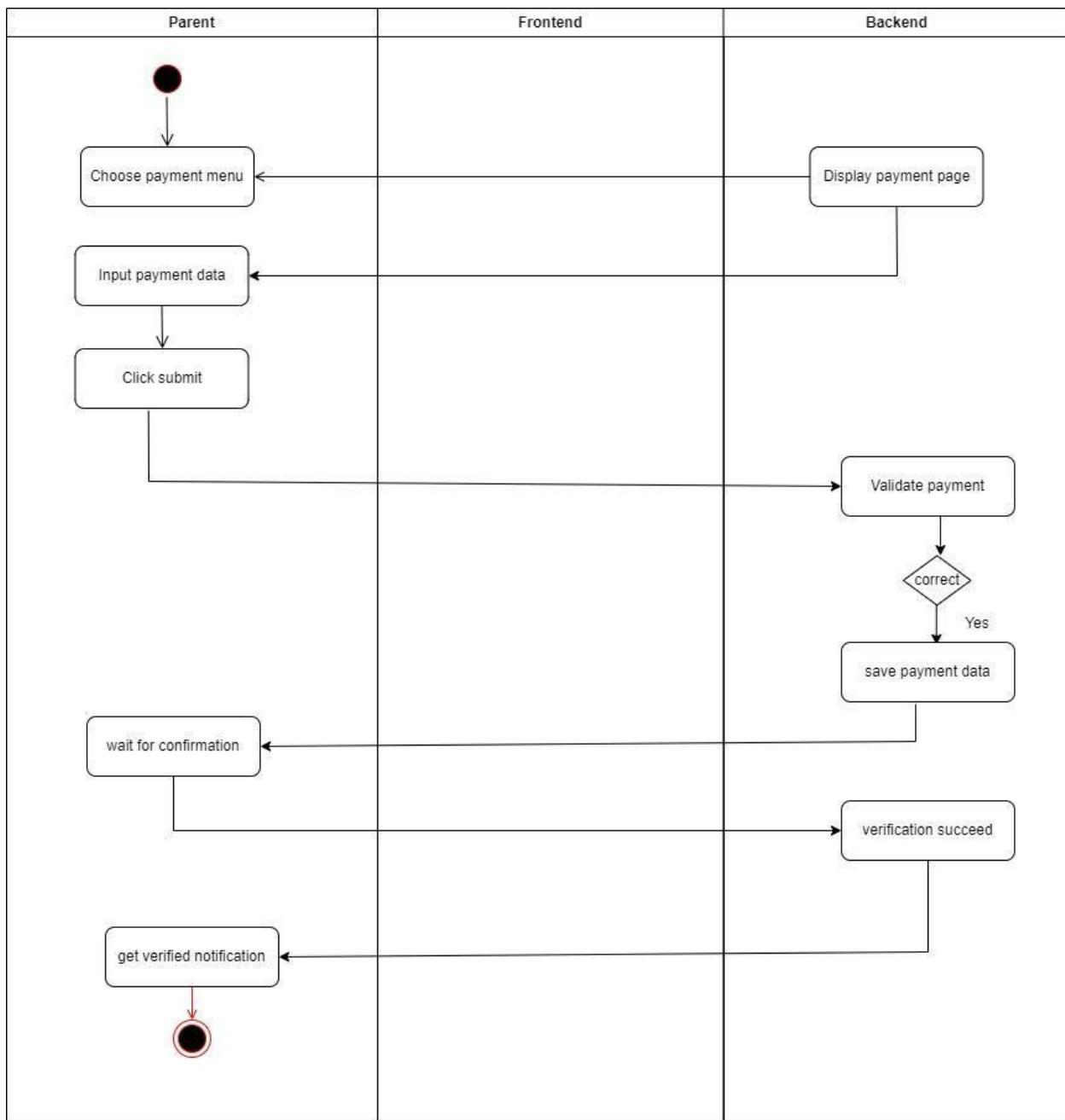


Figure 8: Activity diagram for parent to do payment

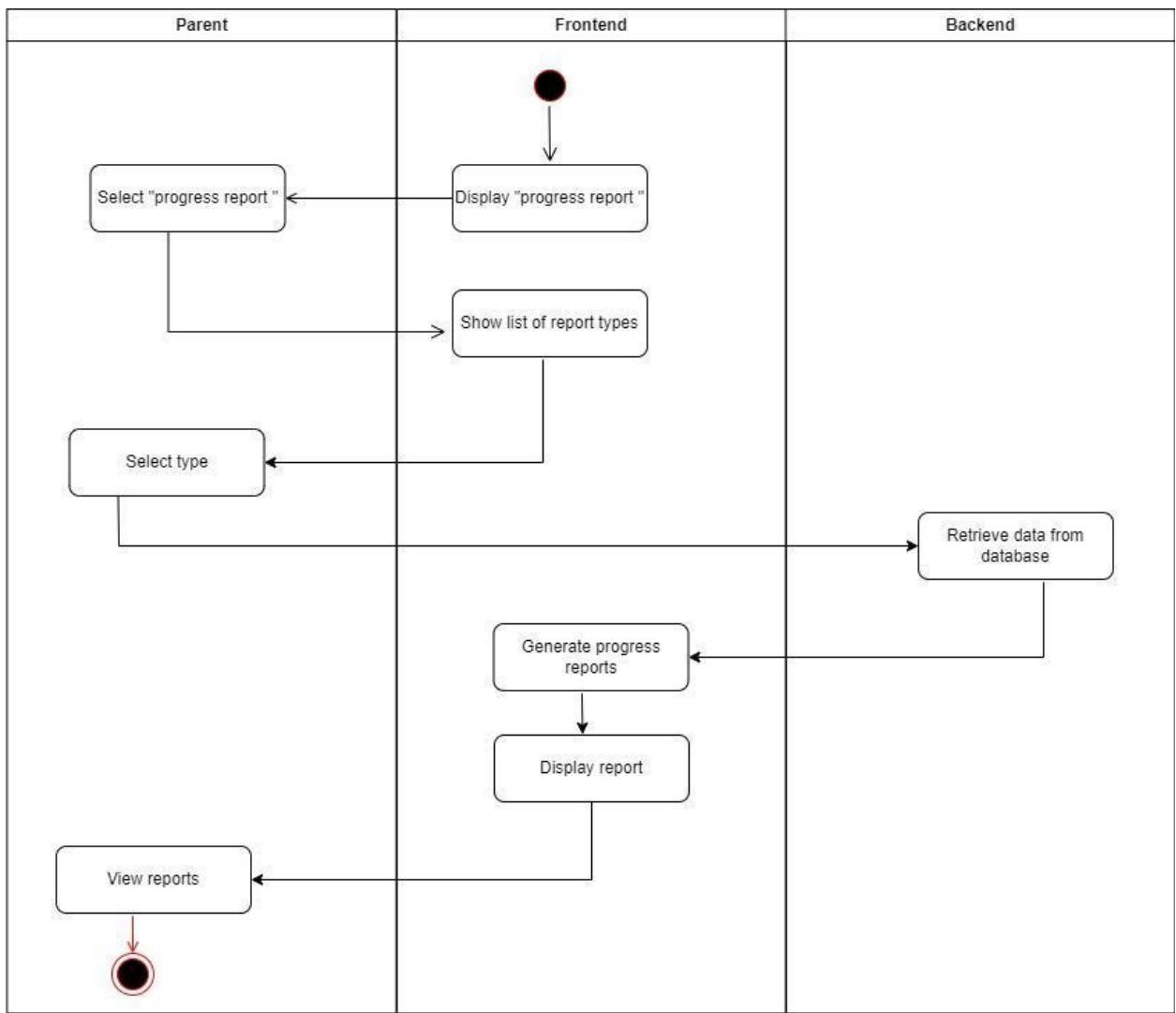


Figure 9: Activity diagram for parent to view progress reports of students

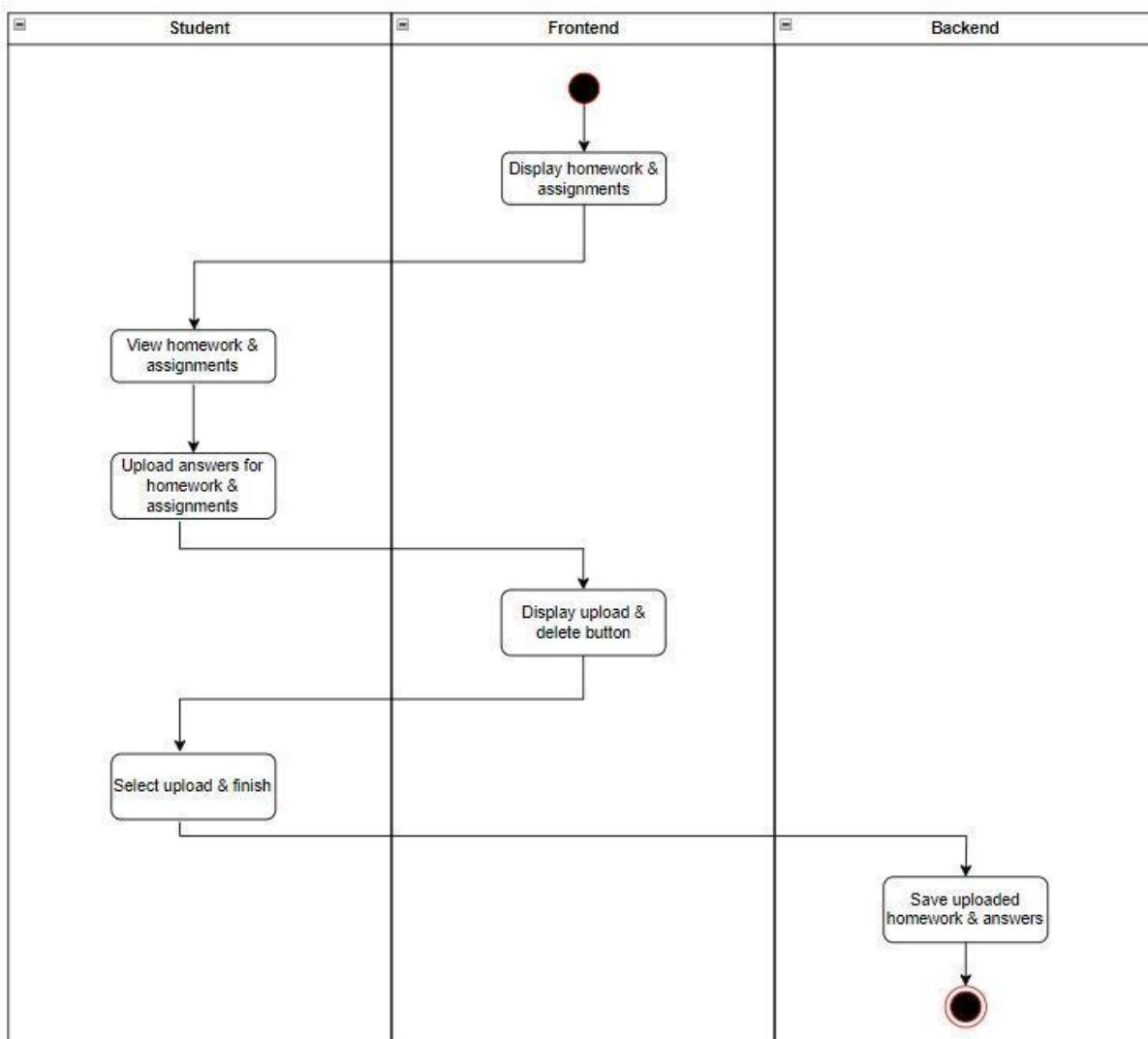


Figure 10: Activity diagram for Student to view homework & assignments

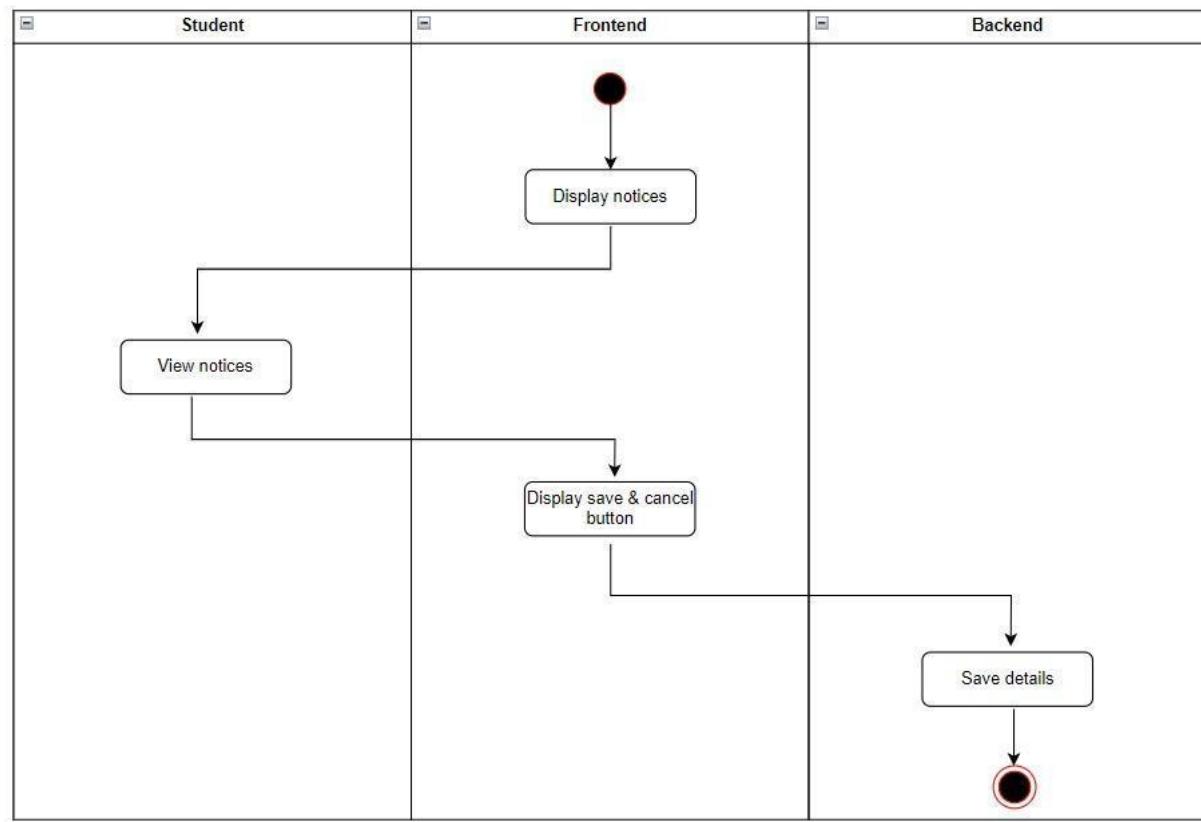


Figure 11: Activity diagram for Student to view notices

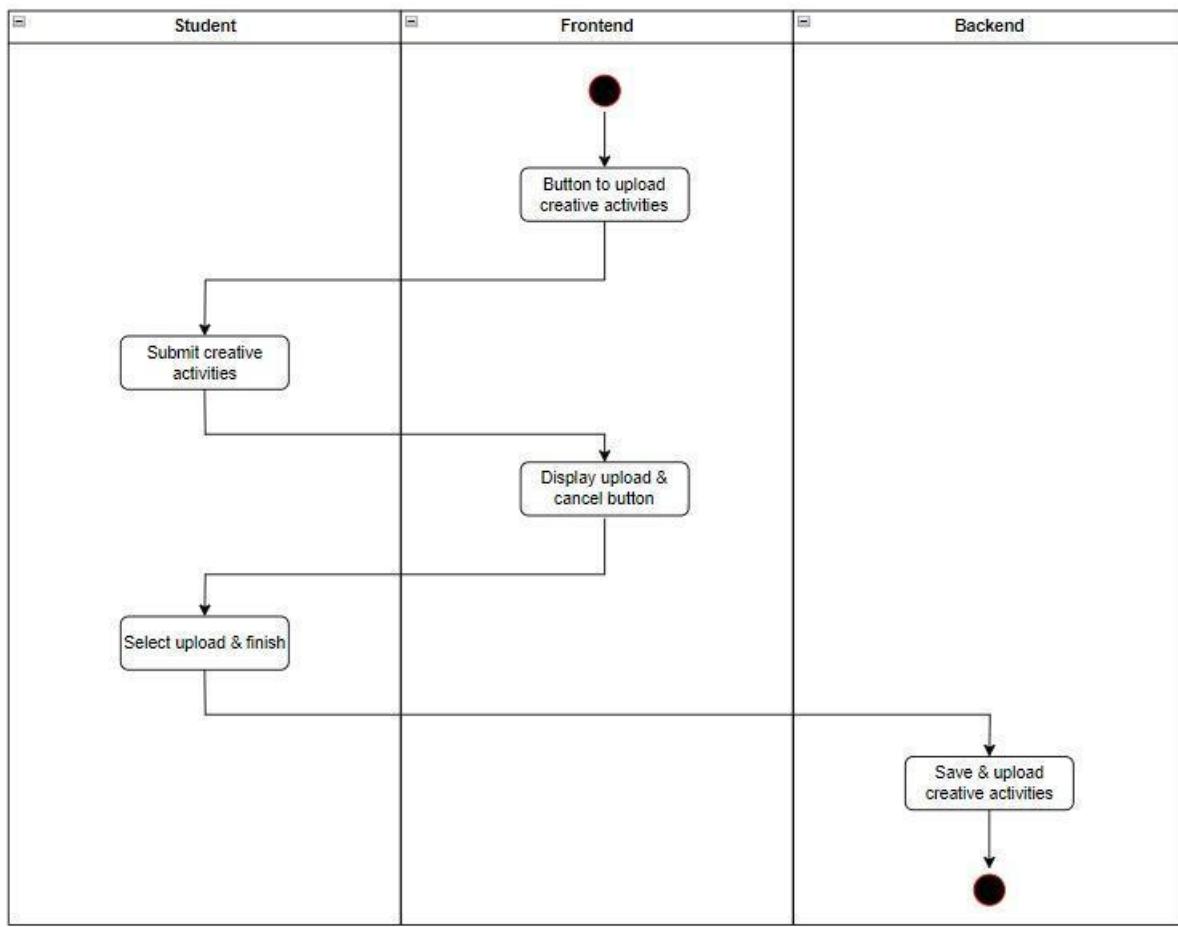


Figure 12: Activity diagram for Student to post creative activities

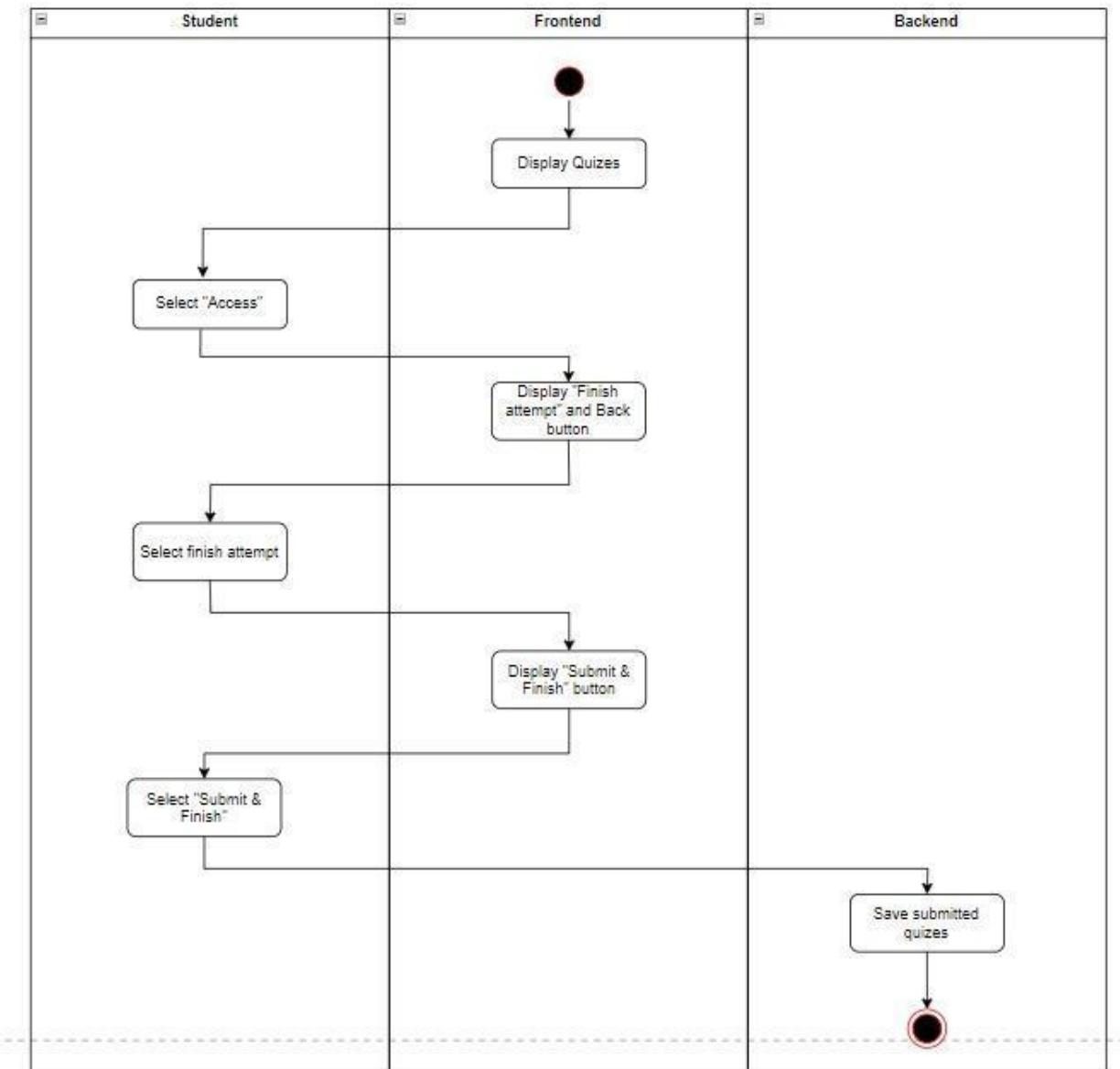


Figure 13: Activity diagram for Student to access to quizzes

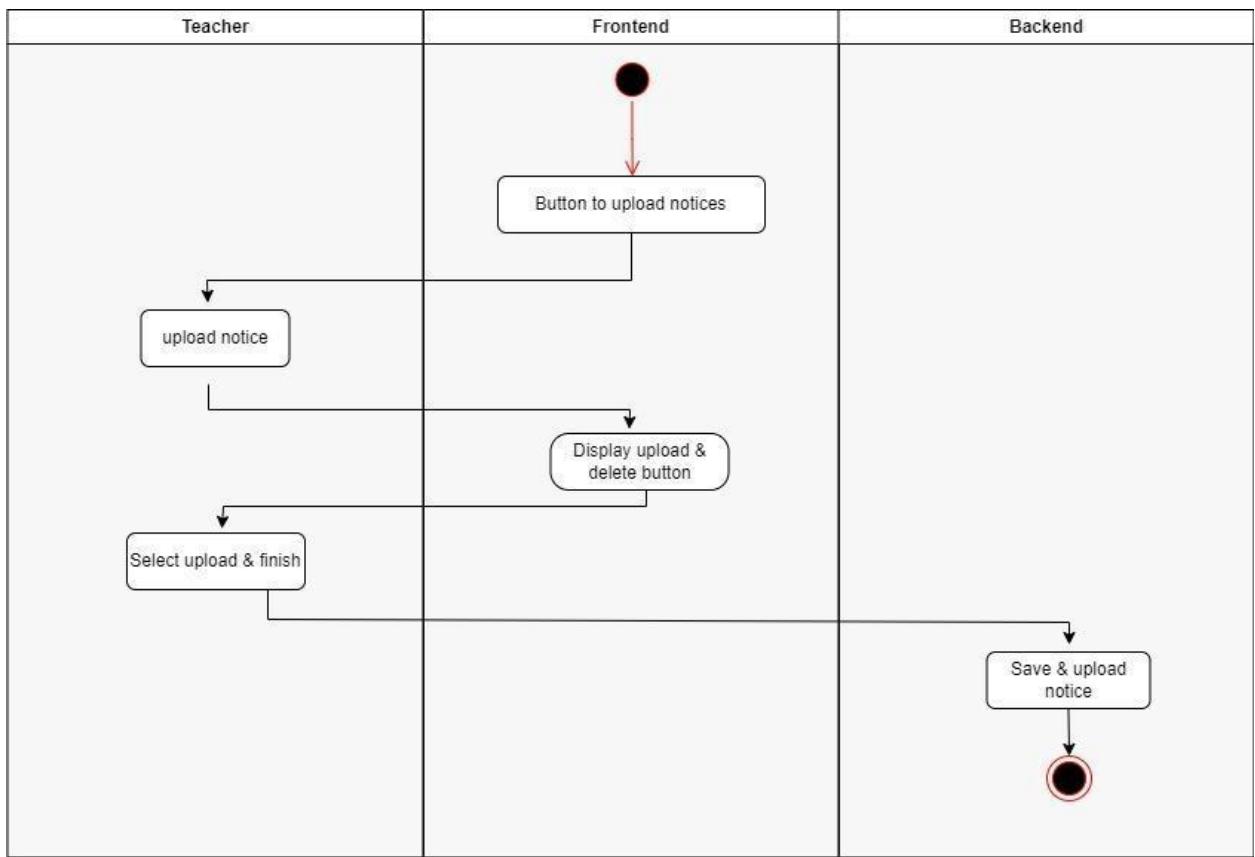


Figure 14: Activity diagram for Teacher to upload notices

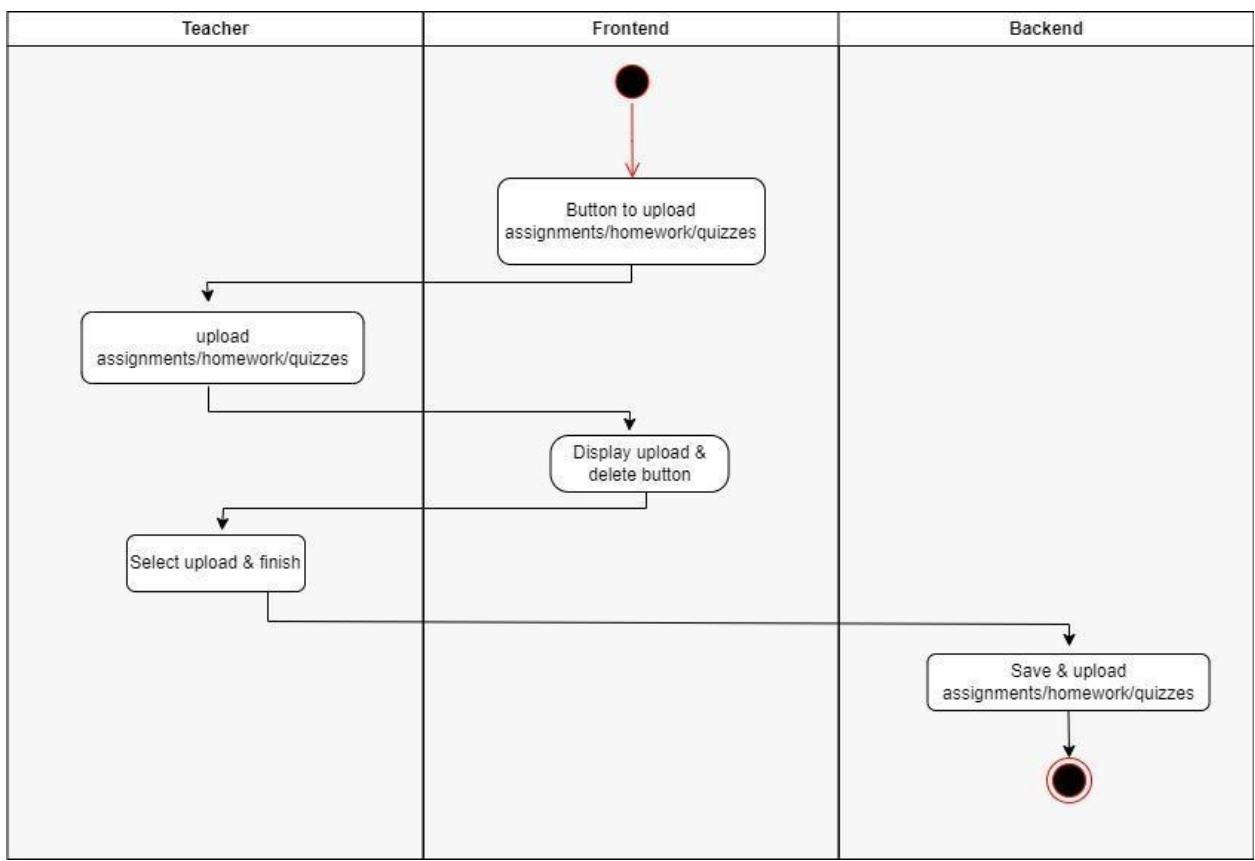


Figure 15: Activity diagram for Teacher to upload assignments/homework/quizzes

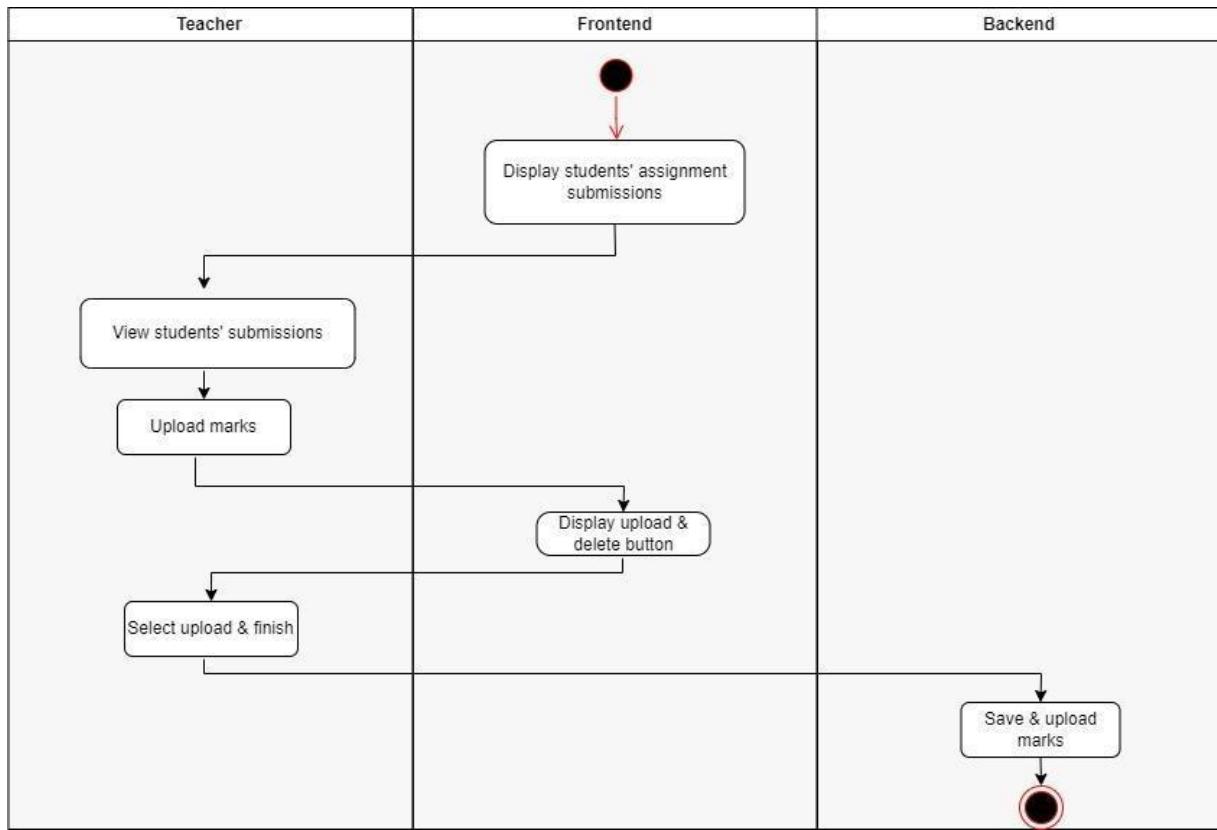


Figure 16: Activity diagram for Teacher to view submission of students

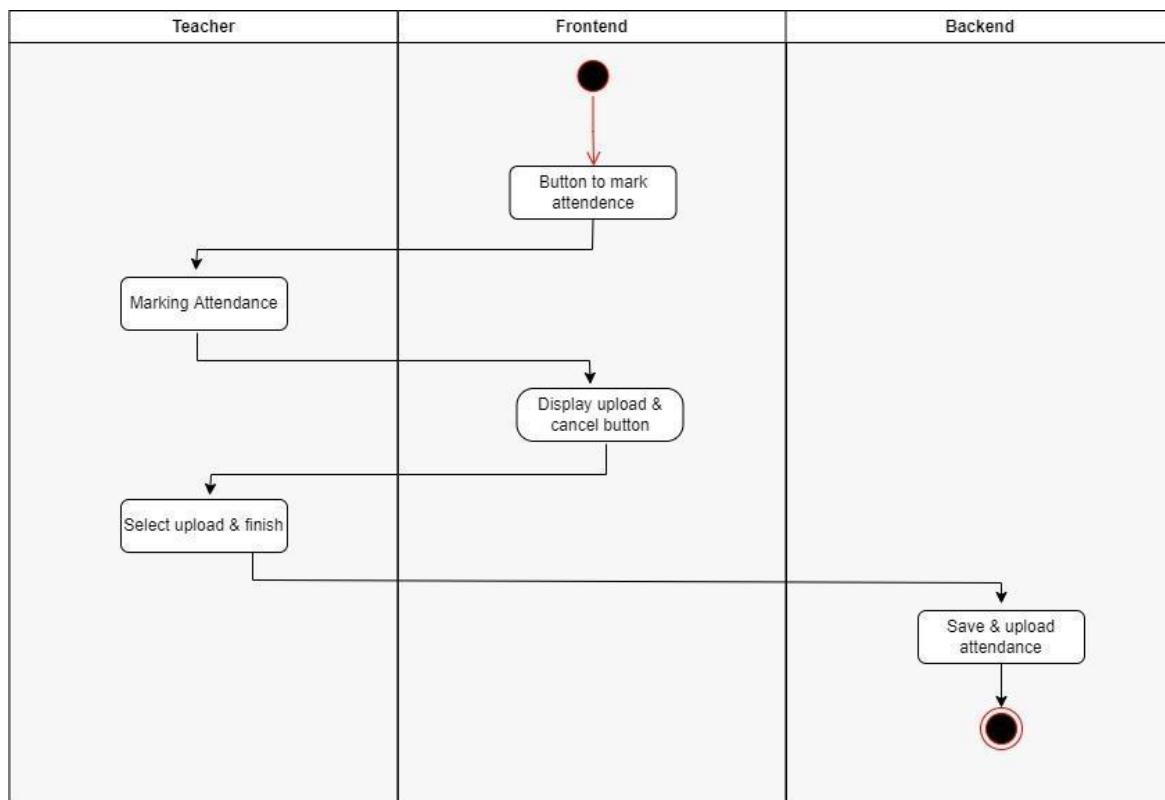


Figure 17: Activity diagram for Teacher to mark attendance

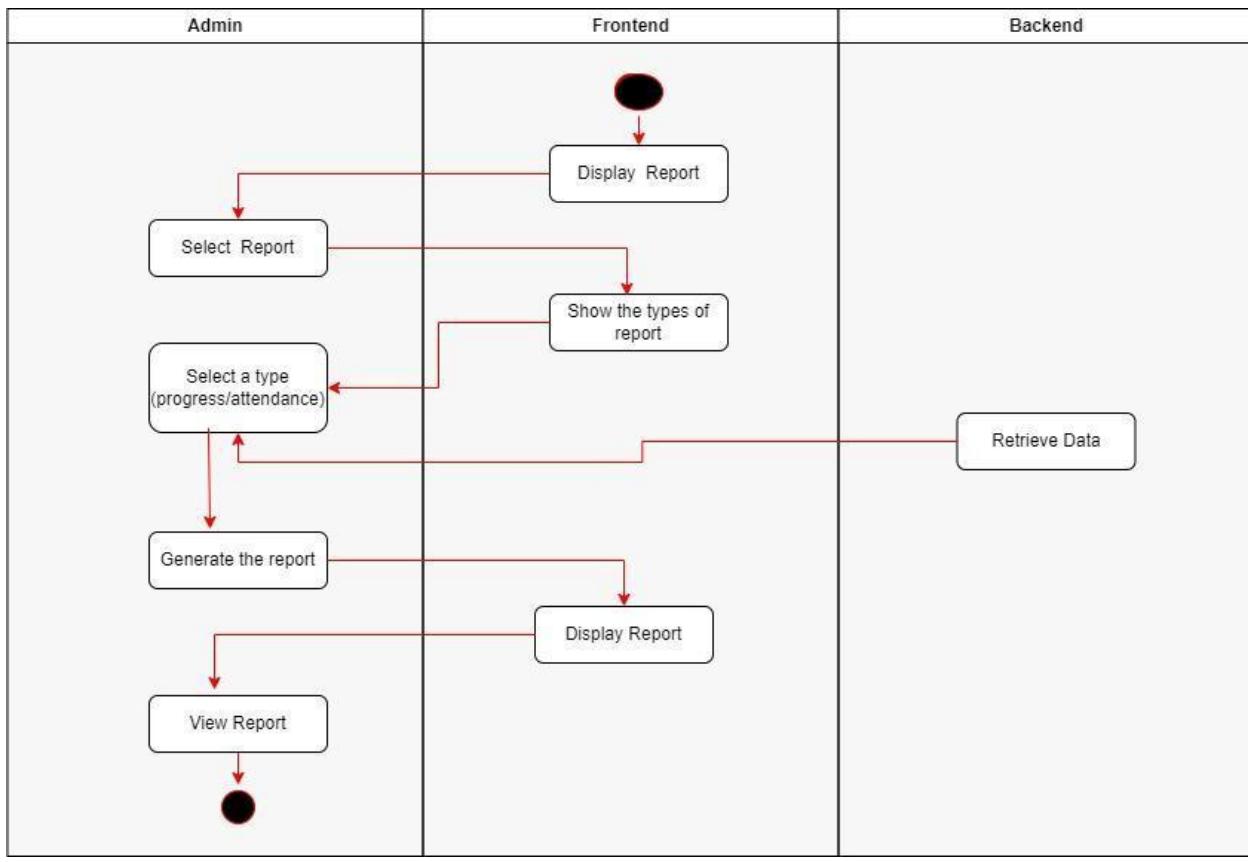


Figure 18: Activity diagram for Admin to generate and view reports

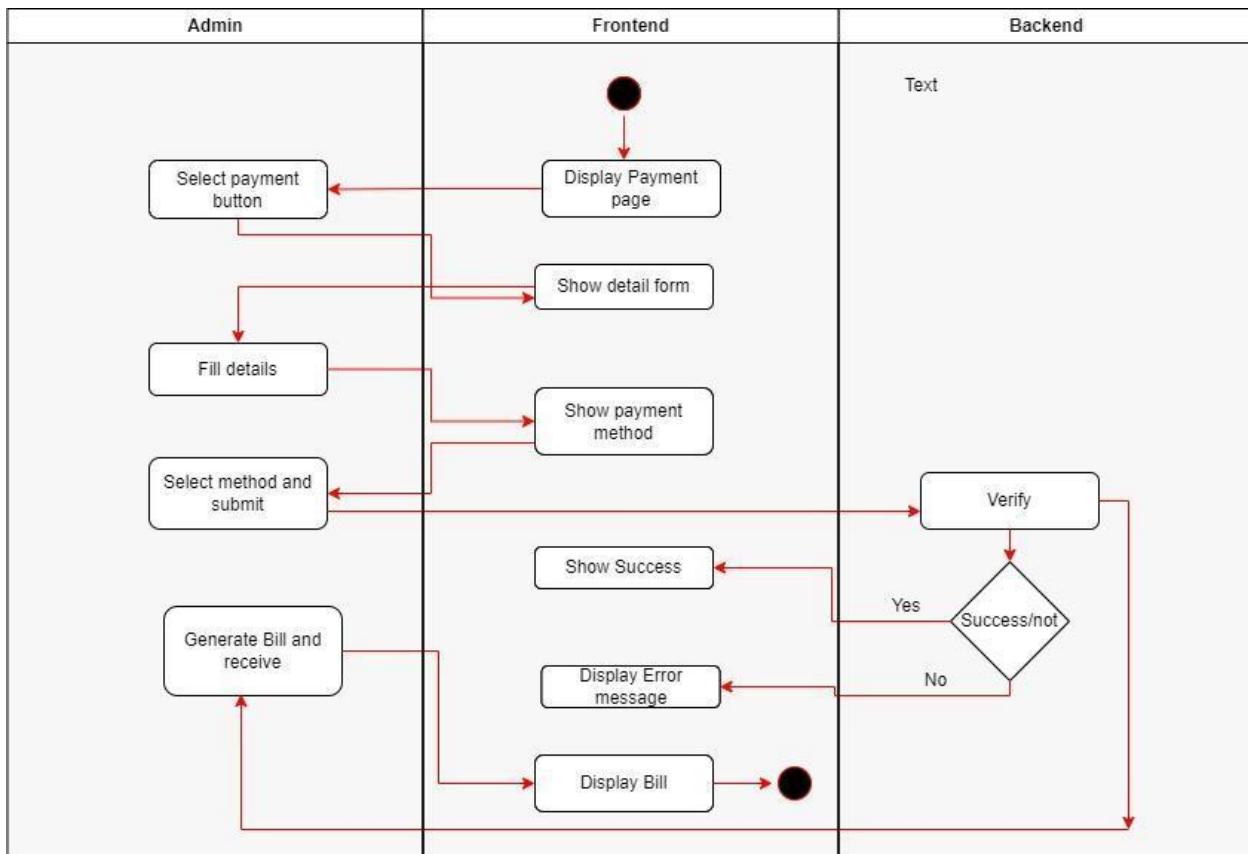


Figure 19: Activity diagram for Admin to handle payments

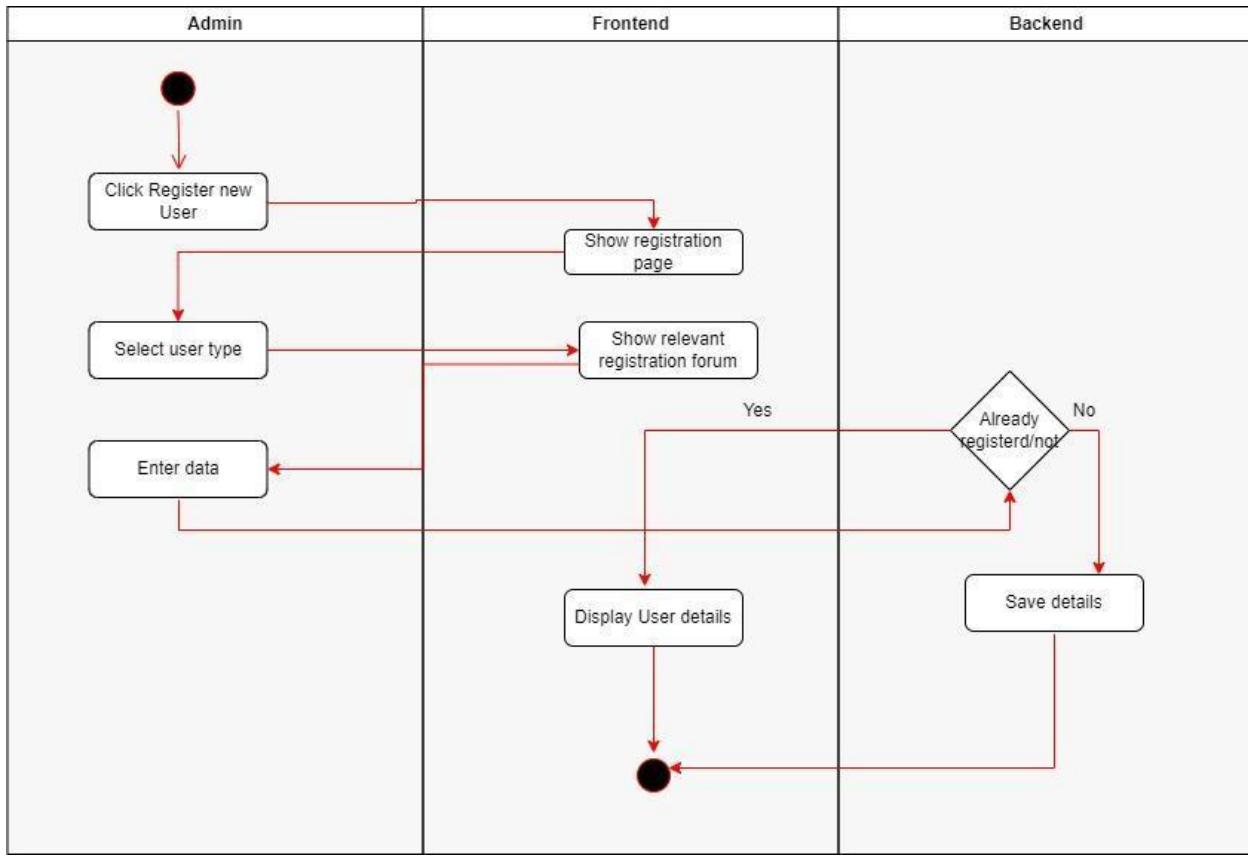


Figure 20: Activity diagram for Admin to handle user registration

5.3.5 Class diagram

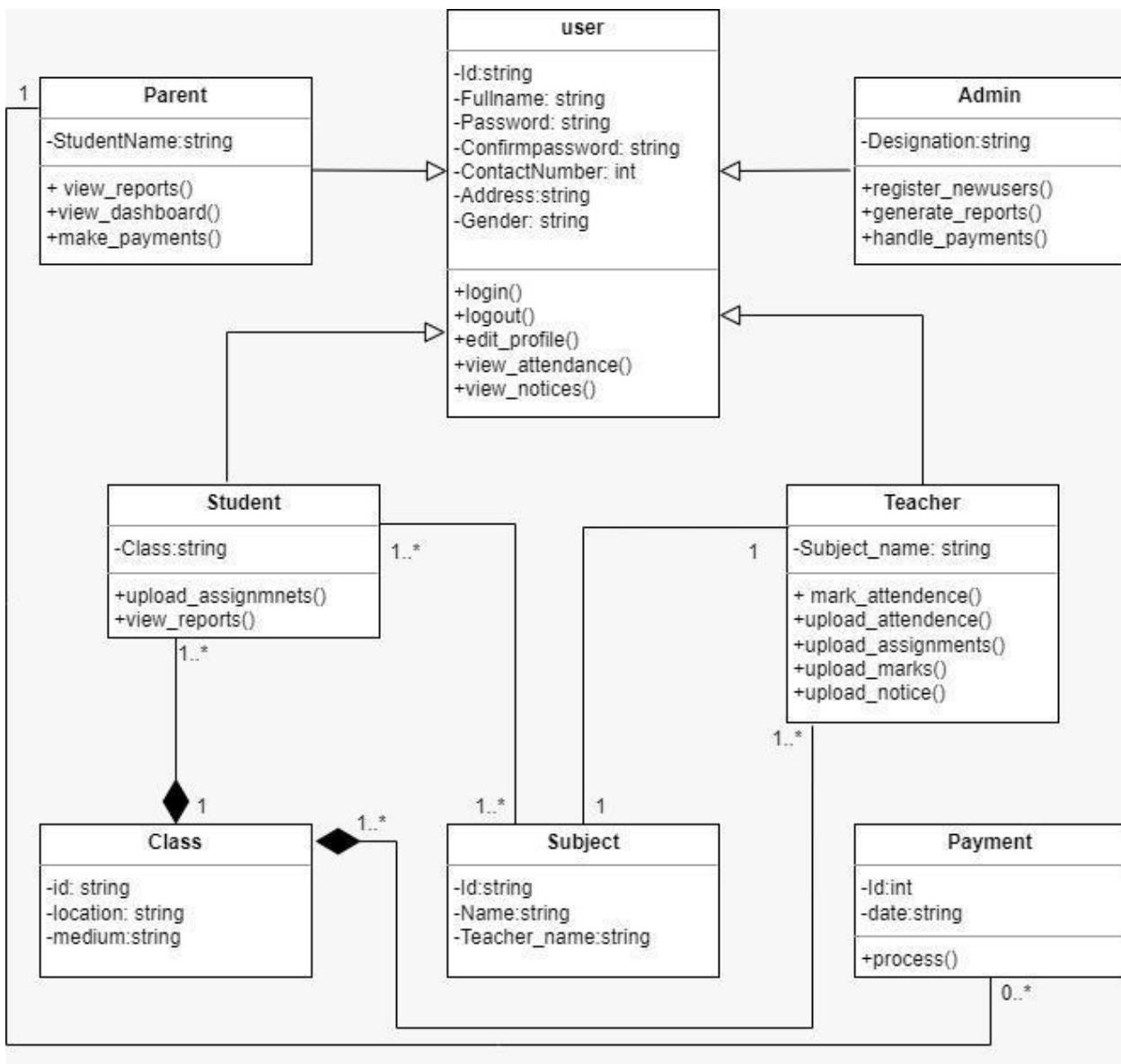


Figure 21: Class diagram

5.3.6 Sequence diagram

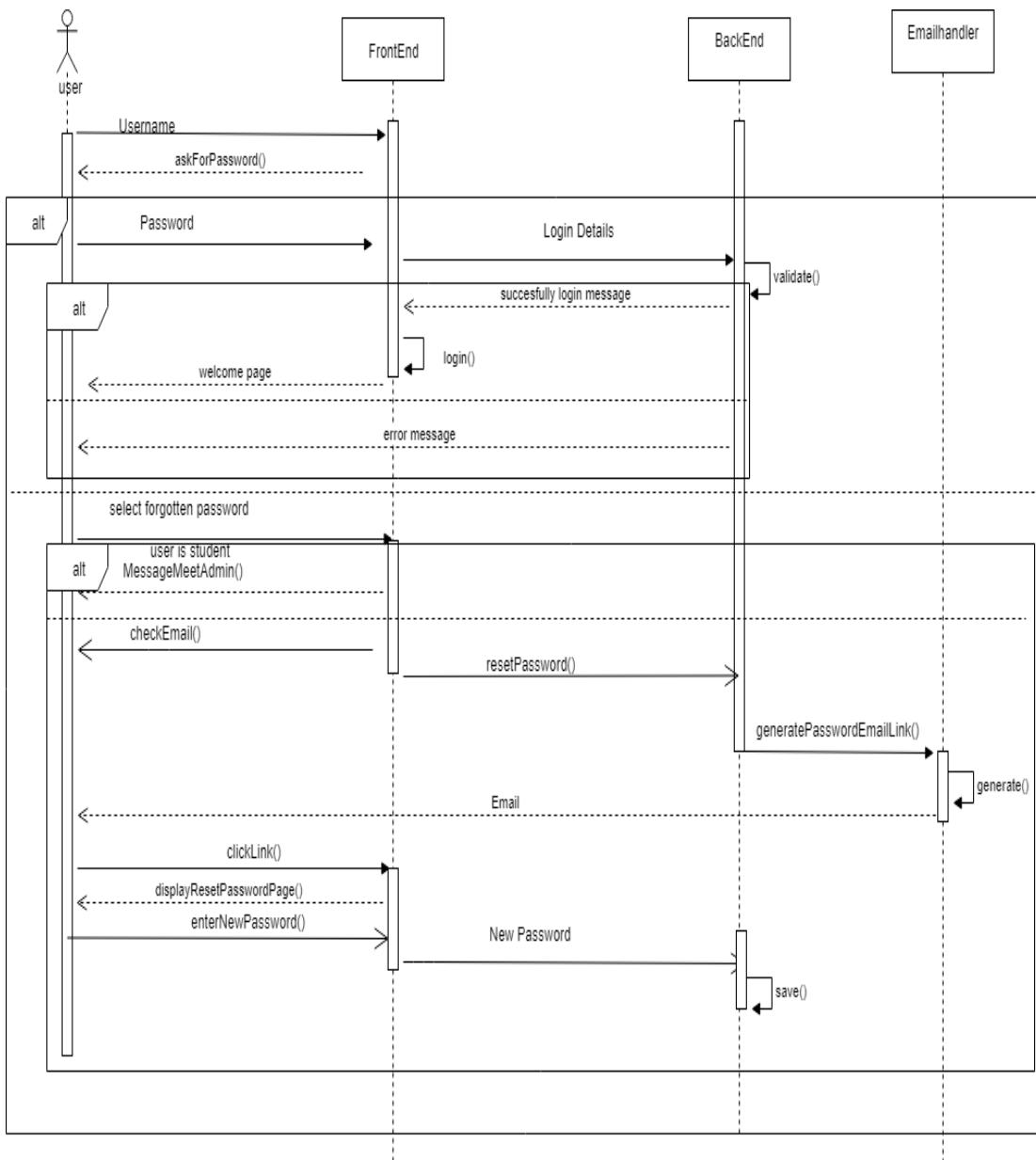


Figure 22: Login and reset password

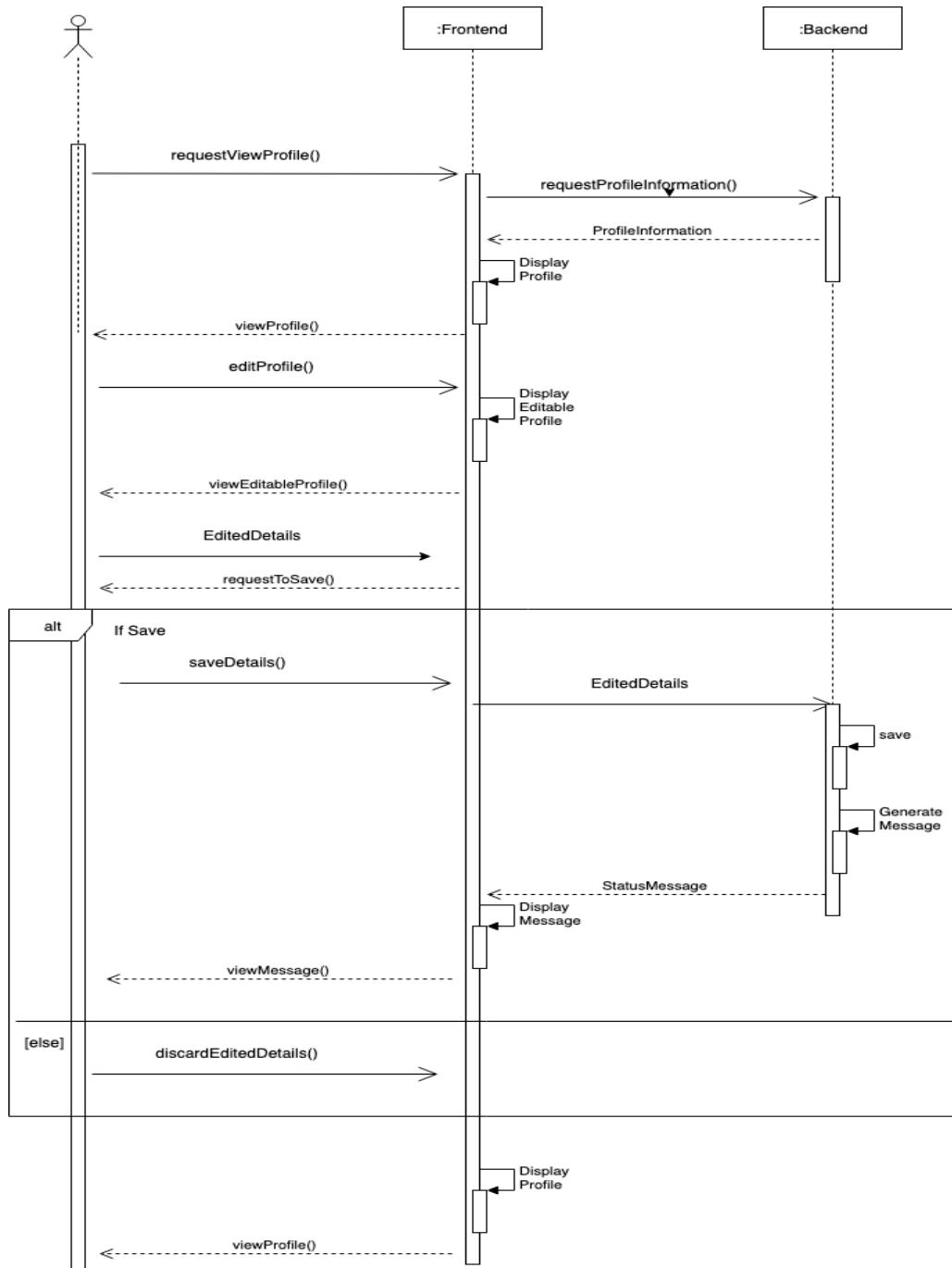


Figure 23: Edit user profile

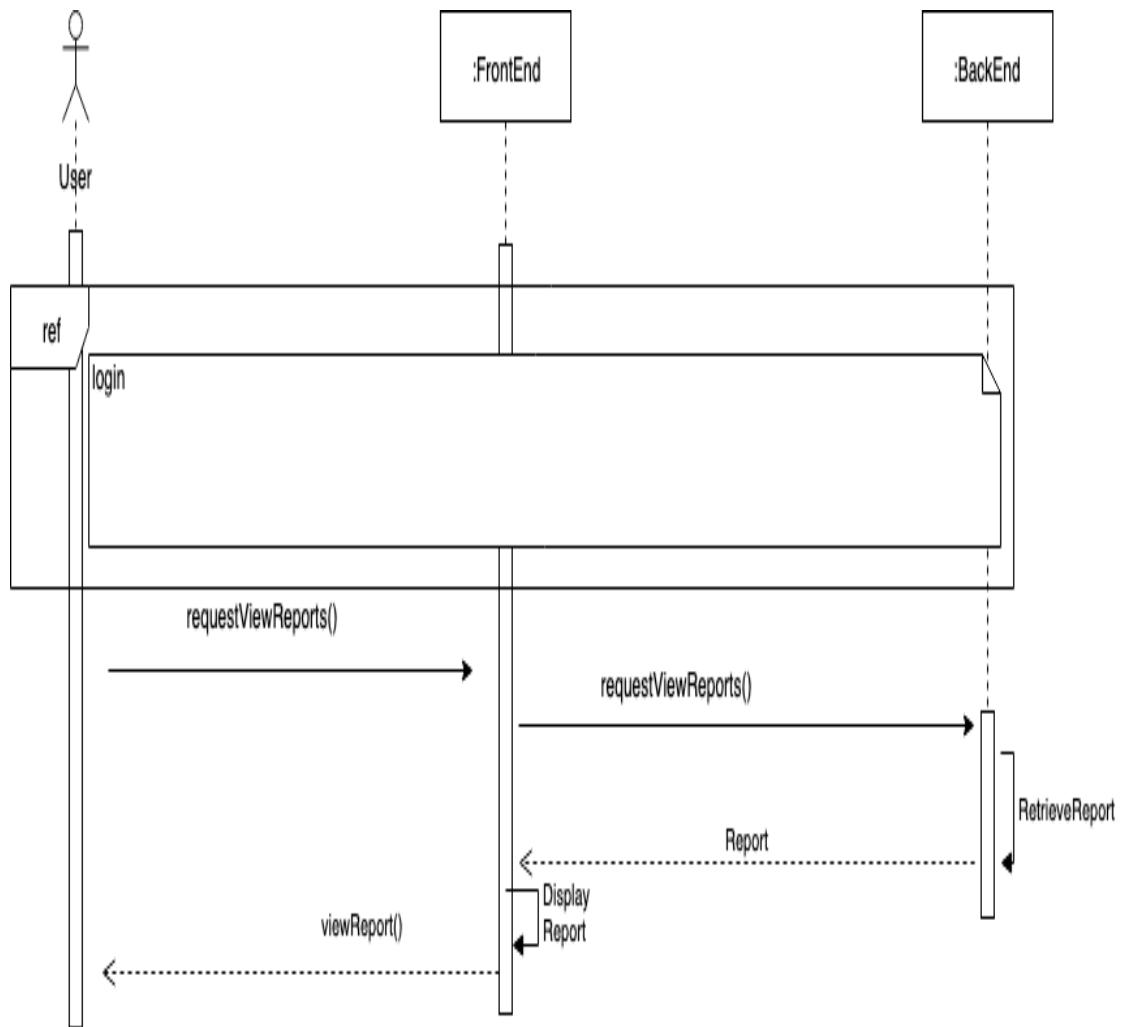


Figure 24: View reports

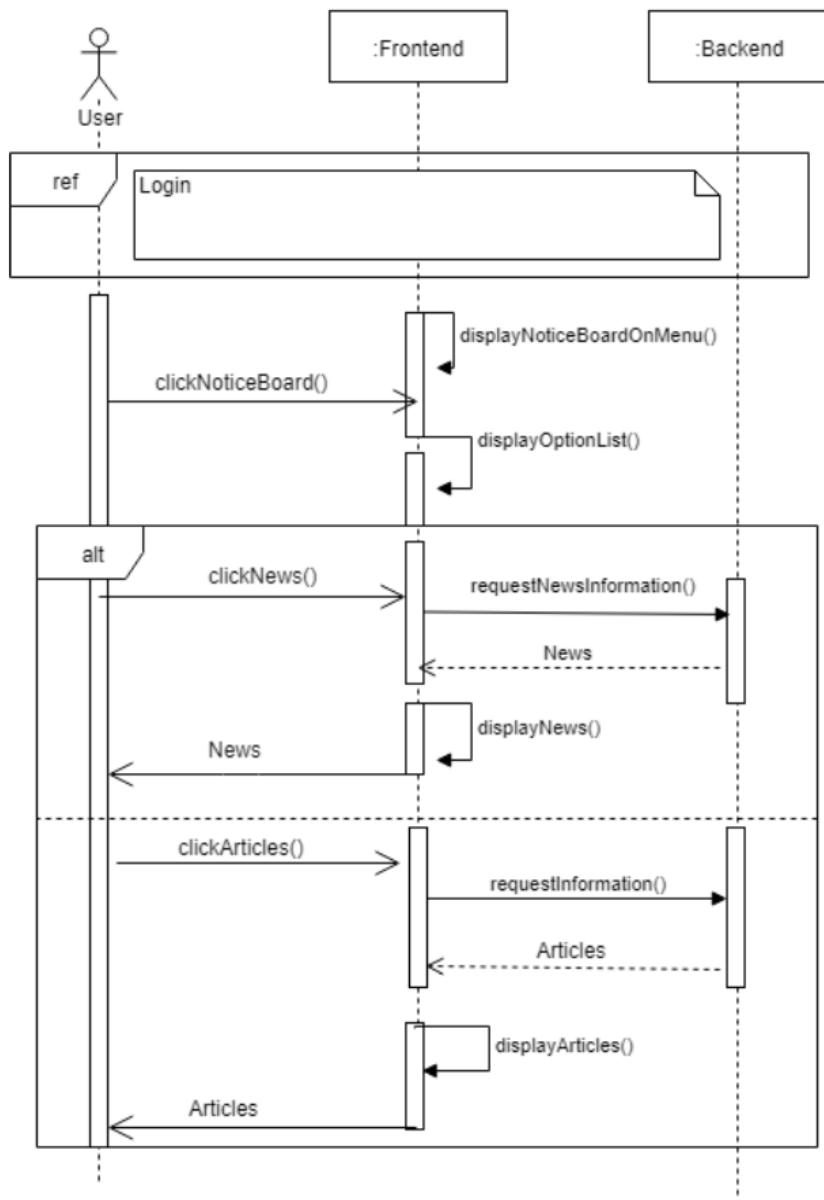


Figure 25: View articles and notices

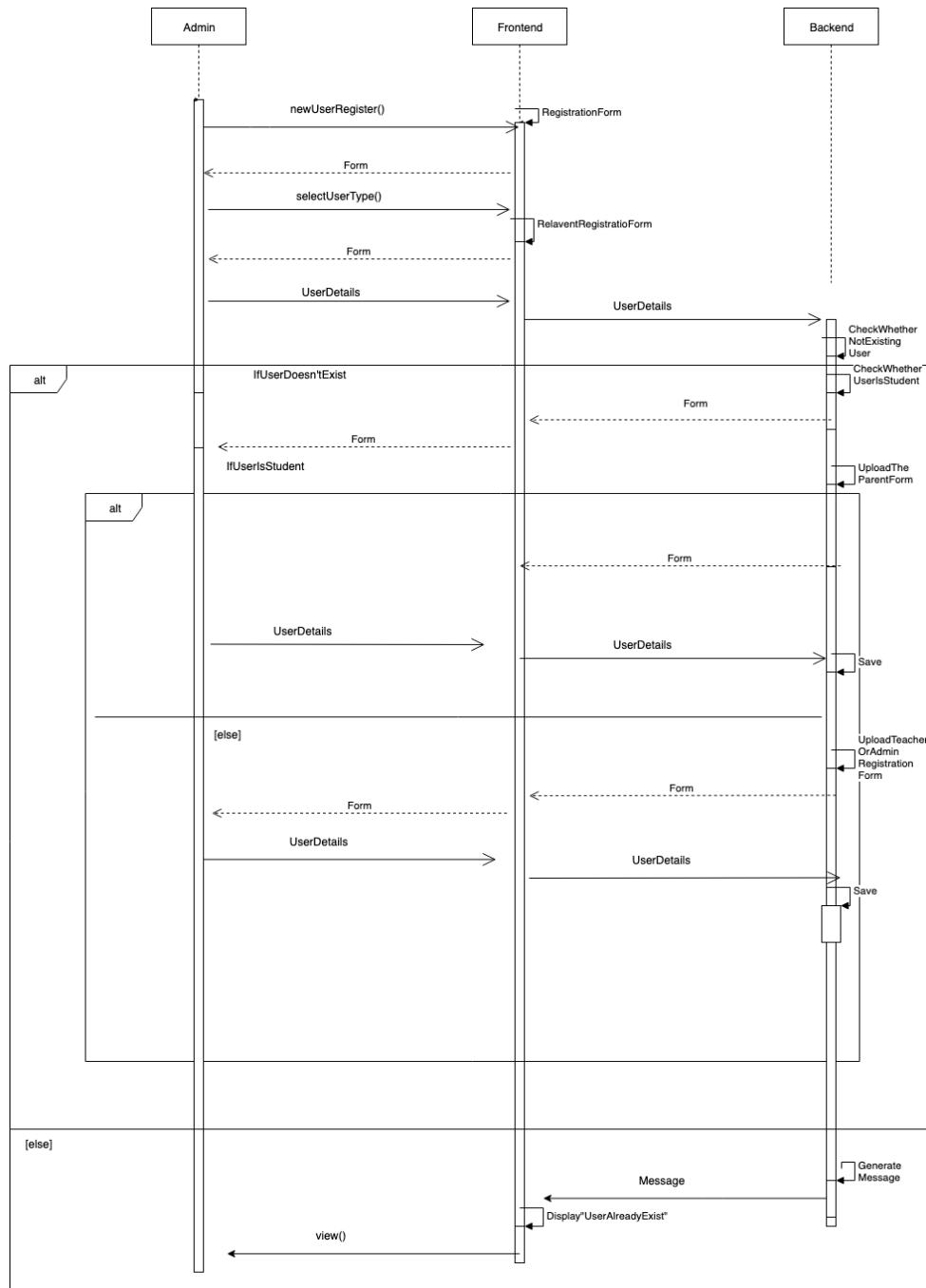


Figure 26: Register new user

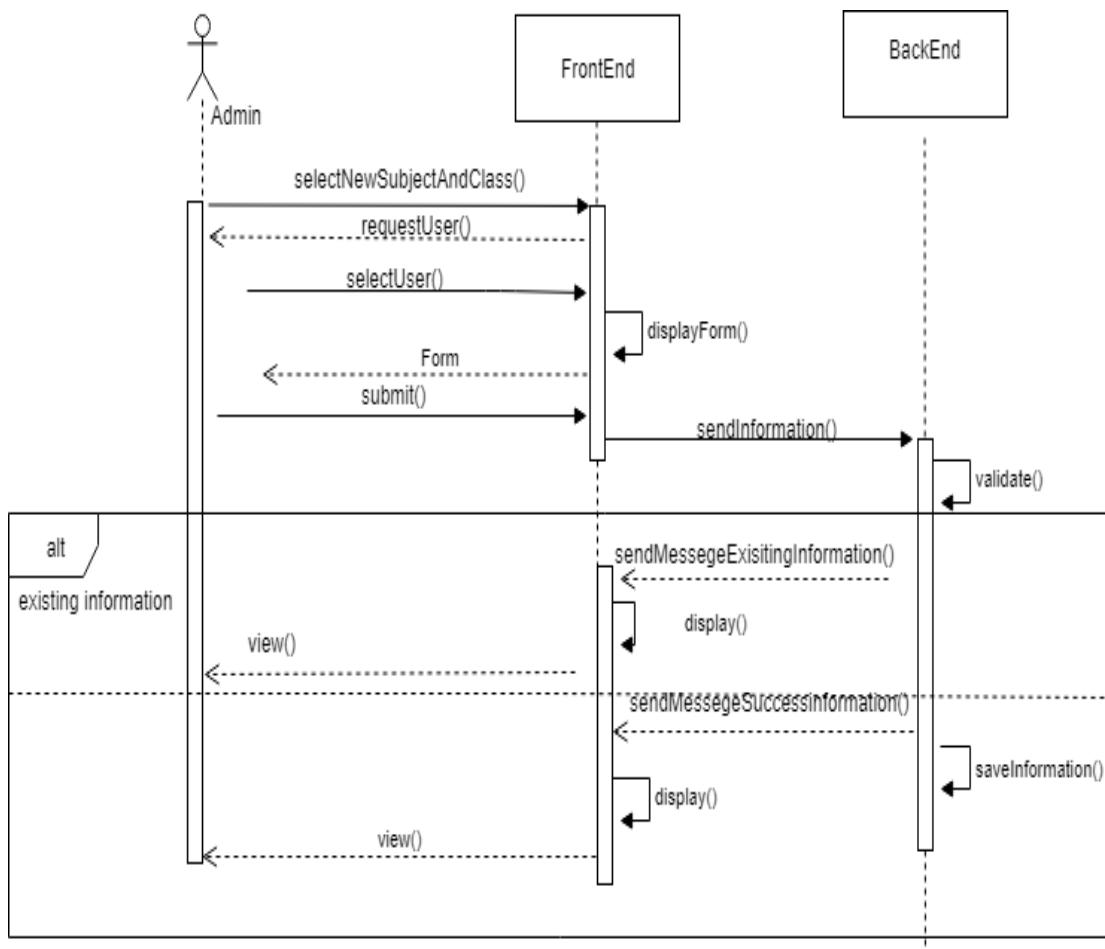


Figure 27: Assign classes and subjects

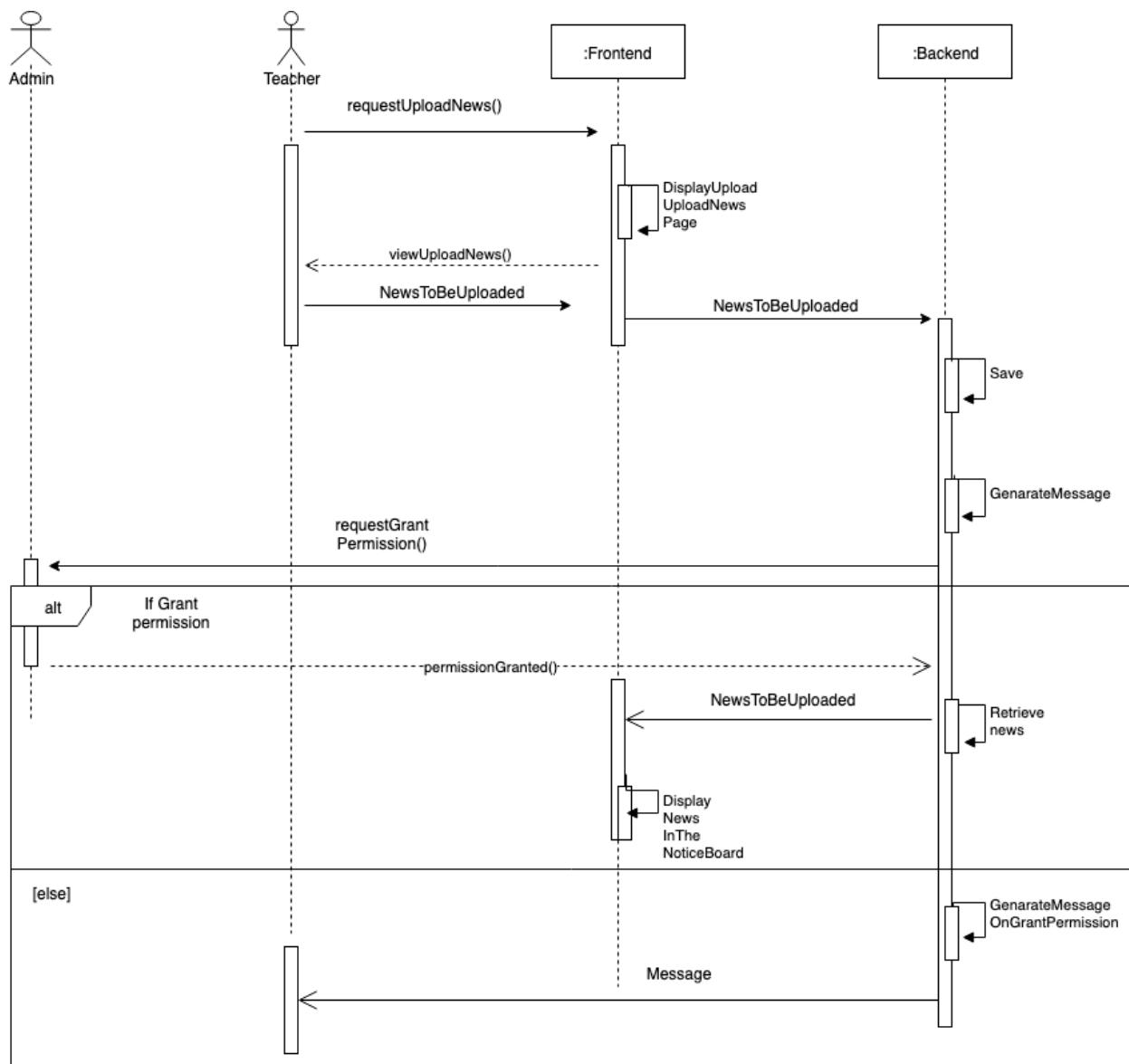


Figure 28: Upload activities

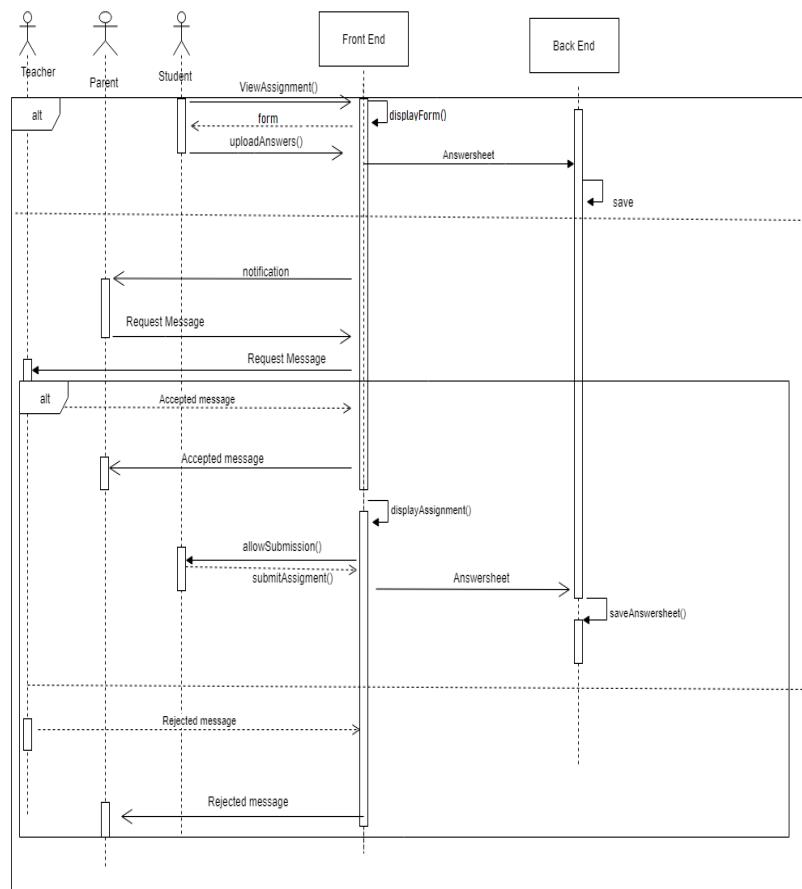


Figure 29: View assignment and upload answers

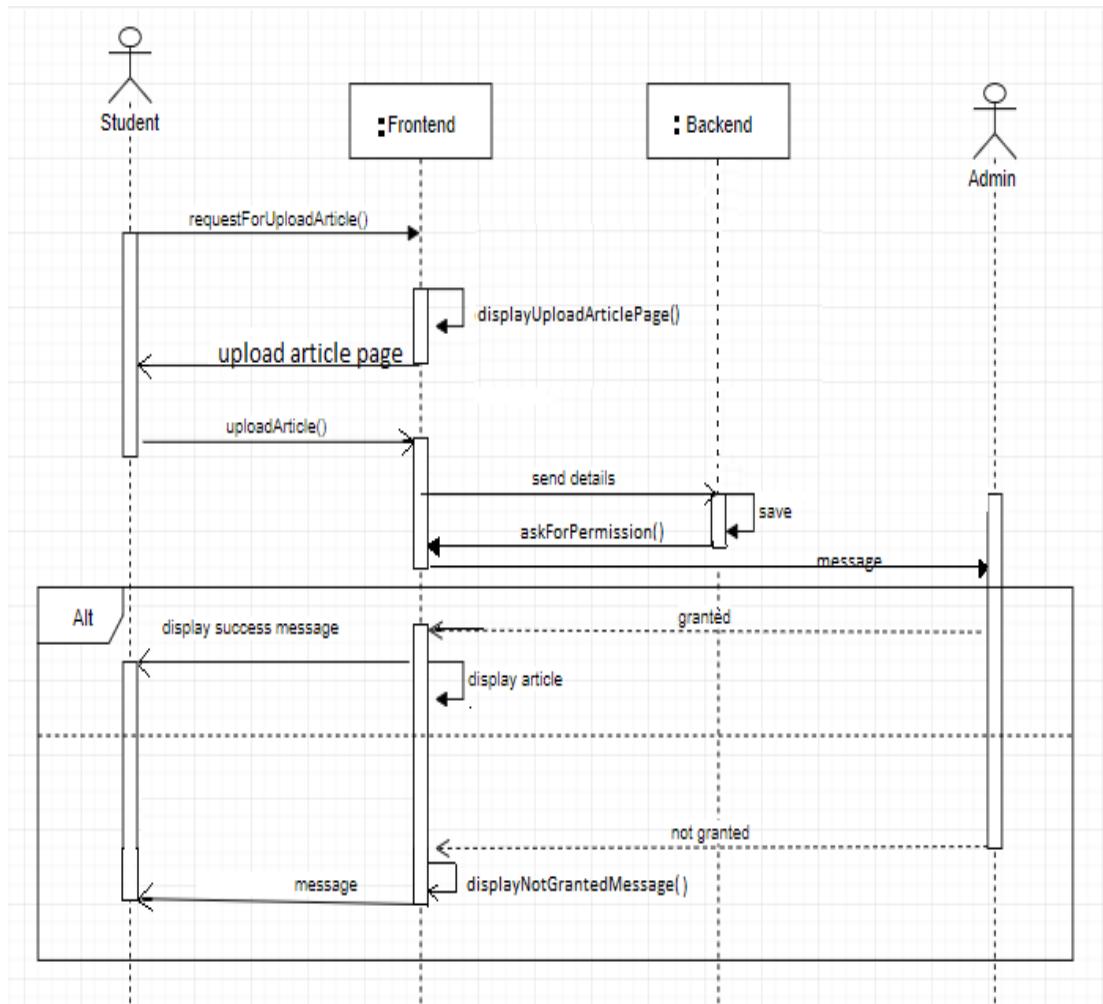


Figure 30: Upload articles

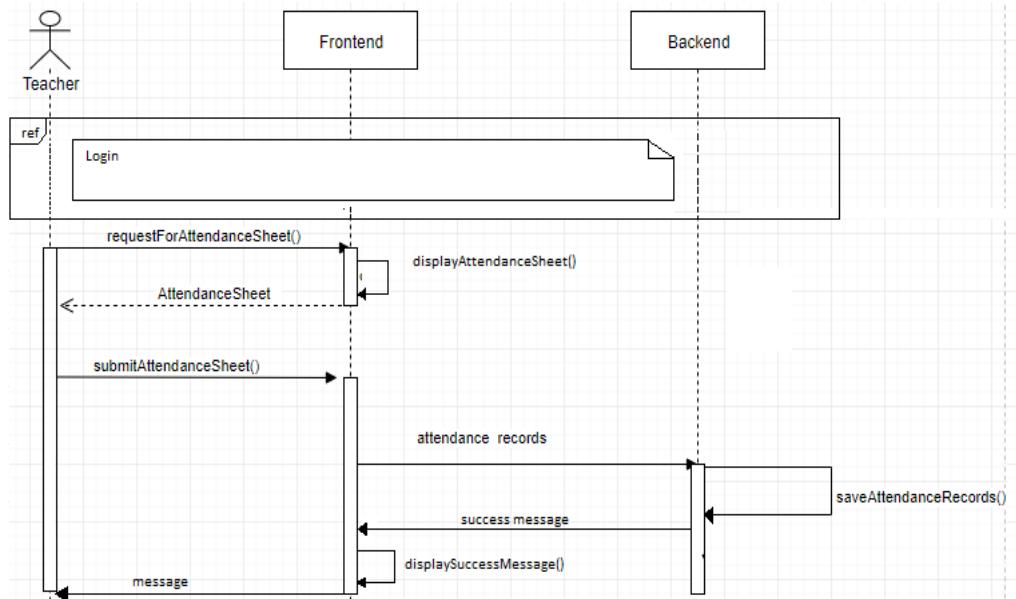


Figure 32: Upload learning materials

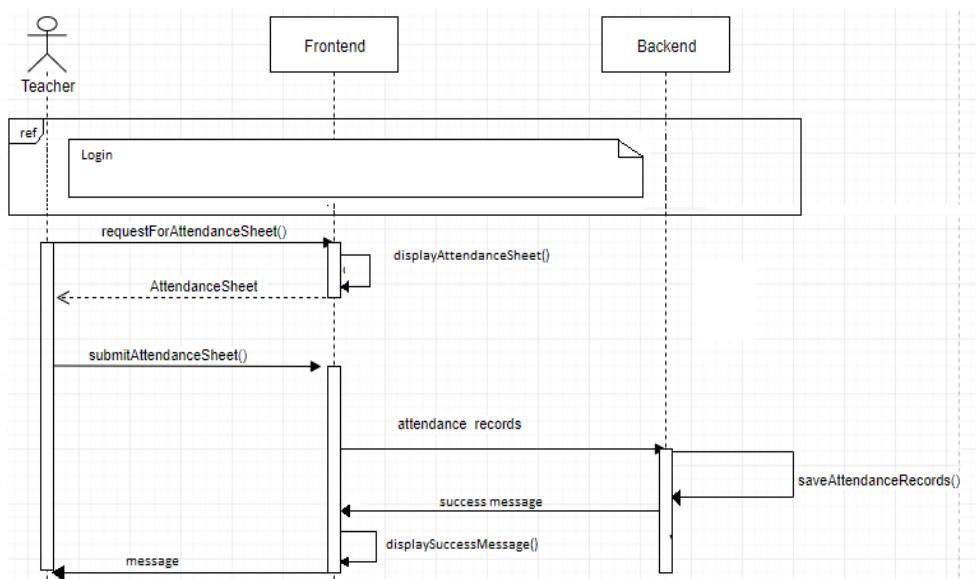


Figure 33: Upload news

5.4 Summary

In the UML diagrams and the EER diagram it shows design details of our system. In the EER diagram it shows what the entities in the database are and how they are related. Use case diagram shows all the requirements from user's point of view. Activity diagram shows flow of activities in the system. Class diagram shows the classes and their relationships. Sequence diagram shows timebased dynamics of interactions

Chapter 5

Implementation

5.1 Introduction

After identifying user requirements, we had a meeting with our client and we modified some user requirements. We selected the agile scrum as our software process model. First, we selected the technologies that we can use to fulfil the user requirements and had a knowledge about those technologies. Then we drew the UML diagrams and EER diagram in order to analyze user requirements. Also, we have drew use case diagram, class diagram, activity diagrams and sequence diagrams which are regarding to our system. Then we corrected those diagrams according to the comments given by our supervisor. Our system consists of two main modules as web application and mobile application. Currently we are designing the database and part of web application.

5.1.1 Interfaces

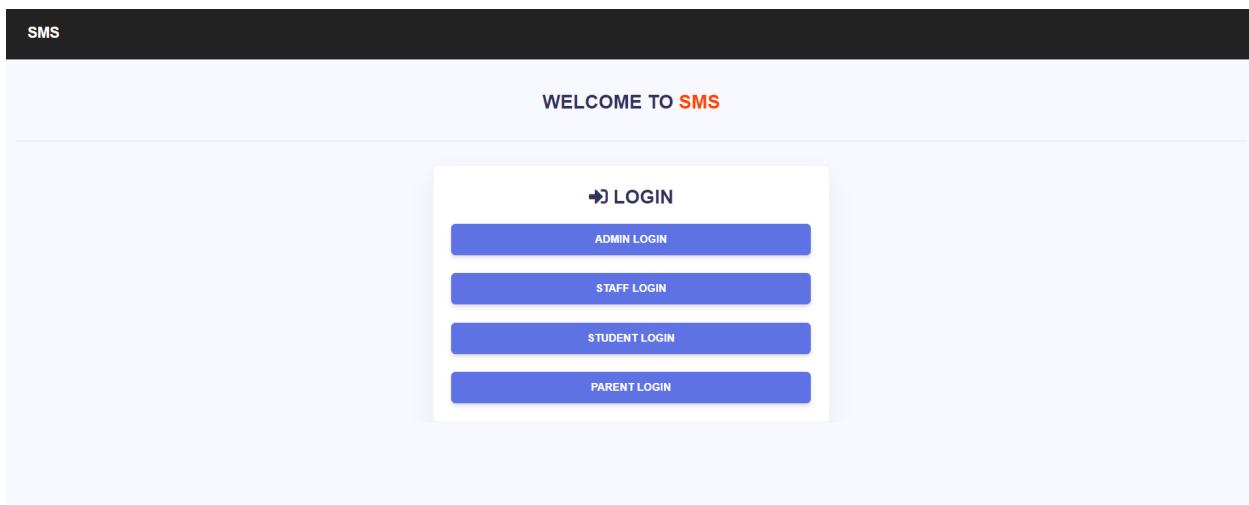
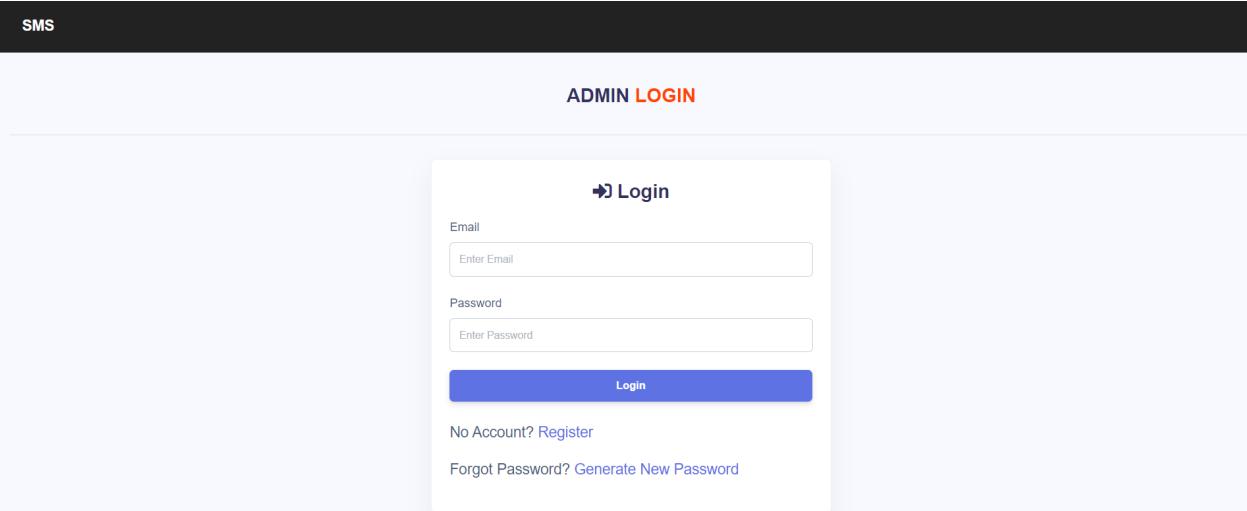


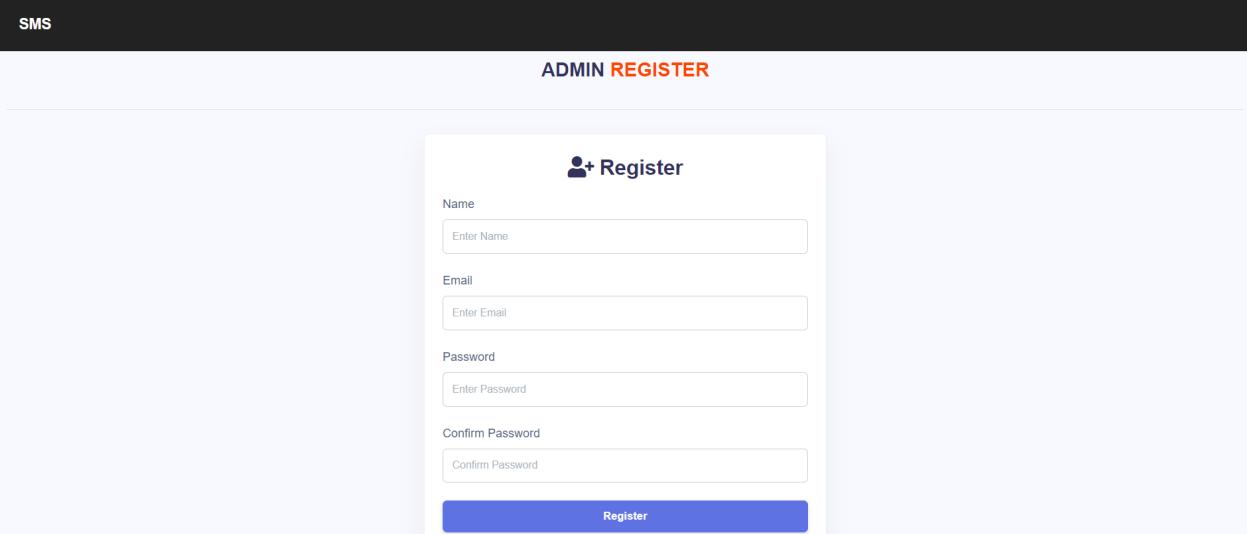
Figure 34: UI for user login

5.1.1. Admin Module



The screenshot shows the 'ADMIN LOGIN' interface. At the top, there is a black header bar with the text 'SMS'. Below it, the main title 'ADMIN LOGIN' is centered. A central modal window titled 'Login' contains fields for 'Email' and 'Password', each with an 'Enter Email' placeholder. Below the fields is a blue 'Login' button. At the bottom of the modal, there are two links: 'No Account? Register' and 'Forgot Password? Generate New Password'.

Figure 35: UI for Admin login



The screenshot shows the 'ADMIN REGISTER' interface. At the top, there is a black header bar with the text 'SMS'. Below it, the main title 'ADMIN REGISTER' is centered. A central modal window titled 'Register' contains fields for 'Name', 'Email', 'Password', and 'Confirm Password', each with an 'Enter Name', 'Enter Email', 'Enter Password', and 'Confirm Password' placeholder respectively. Below the fields is a blue 'Register' button.

Figure 36: UI for Admin registration

SMS

Forgot your password?

Change your password in three easy steps. This will help you to secure your password!

1. Enter your email address below.
2. Our system will send you a temporary link
3. Use the link to reset your password

Enter your email address

Enter the email address you used during the registration. Then we'll email a link to this address.

Get New Password

[Back to Login](#)

Figure 37: UI for Forgot password

SMS

Hello Dilki



Manage Staff

Manage your staff here. You can edit and can add or remove a staff. All your changes will be reflected to them as well.

Add **Modify**



Manage Student

Manage your student's profile here (edit/add/remove). All your changes will be reflected to them as well.

Add **Modify**



Manage Parent

Manage parents's profile here (edit/add/remove). All your changes will be reflected to them as well.

Add **Modify**

Figure 38: UI for Admin overview

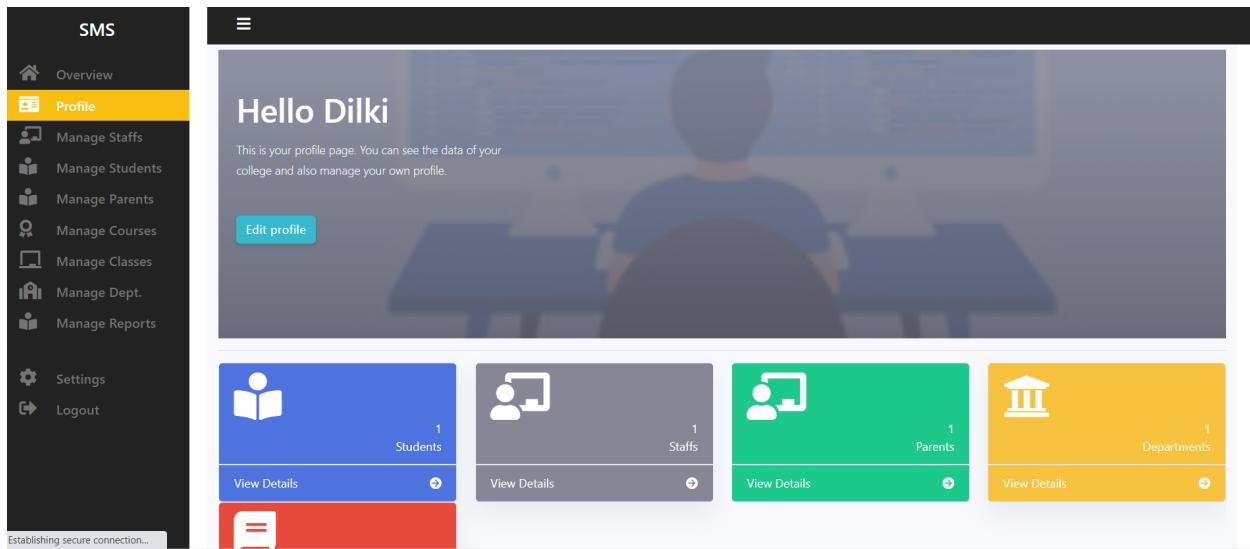


Figure 39: UI for Admin profile

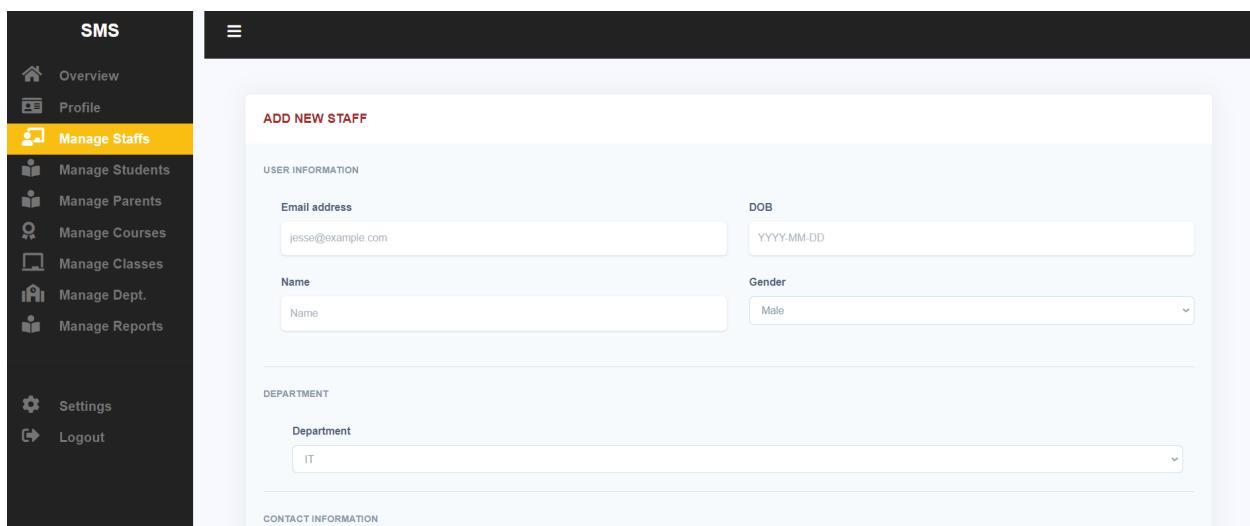


Figure 40: UI for manage profiles

ADD NEW COURSE

COURSE INFORMATION

Course Name	Semester
<input type="text" value="Course Name"/>	<input type="text" value="Semester"/>
Department	
<input type="text" value="IT"/>	
Credits	Course Type
<input type="text"/>	<input type="text" value="Practical"/>

ADD COURSE

Figure 41: UI for manage courses

ALL CLASSES

DEPARTMENT INFORMATION

#	Course	Staff	Section	Manage Class
1	Course A	Staff A	Section A	Manage Class

Figure 42: UI for manage classes

5.1.2. Student Module

The screenshot shows a 'STUDENT LOGIN' page. At the top, there is a dark header bar with the text 'SMS'. Below it, the main title 'STUDENT LOGIN' is centered. A central modal window titled 'Login' contains fields for 'Email' (with placeholder 'Enter Email') and 'Password' (with placeholder 'Enter Password'). Below these fields is a blue 'Login' button. Underneath the button, there are two links: 'Forgot Password?' and 'Generate New Password'.

Figure 43: UI for student login

The screenshot shows a 'Forgot your password?' page. At the top, there is a dark header bar with the text 'SMS'. The main title 'Forgot your password?' is centered above a brief description: 'Change your password in three easy steps. This will help you to secure your password!'. Below this, a numbered list of steps is provided: 1. Enter your email address below., 2. Our system will send you a temporary link., 3. Use the link to reset your password. A large input field labeled 'Enter your email address' is present, with a note below it: 'Enter the email address you used during the registration. Then we'll email a link to this address.' Below the input field is a blue 'Get New Password' button. At the bottom of the page is a red 'Back to Login' button.

Figure 44: UI for student forgot password

Reset your password

Change your password. Make sure it's healthy and safe!

NOTE: The link will expire in 20m

Enter your new password

Enter a new password

Confirm your Password

Confirm password

Change Password

Back to Login

Figure 45: UI for student reset password

The screenshot shows the student overview interface. On the left is a vertical navigation bar with the following items:

- SMS** (highlighted)
- Overview** (selected)
- Profile
- Attendance
- TimeTable
- Marks
- Fee History
- Upload Assignments
- Assignments
- Settings
- Logout

The main content area has a header "Hello Dilki". It features three cards:

- My Attendance**: Shows a red icon with a white person silhouette and a checkmark. Below it is a description: "View the attendance status for each of your courses. (Make sure that the atted. is atleast 75%)". A blue "View Attendance" button is at the bottom.
- Marks**: Shows an icon of a clipboard with a grade "A+". Below it is a description: "View the marks obtained for each of your courses. These include the marks upto your previous semester.". A blue "View Marks" button is at the bottom.
- Time Table**: Shows an icon of a hand writing on a grid. Below it is a description: "View your timetable. (It shows the time table of all the courses you have registered for!)". A blue "View Timetable" button is at the bottom.

Figure 46: UI for student overview

SMS

- Overview
- Profile**
- Attendance
- TimeTable
- Marks
- Fee History
- Upload Assignments
- Assignments
- Settings
- Logout

My account

STUDENT INFORMATION

Student Name	Email
Dilki	dilki.sachinika@gmail.com

Date of Birth	Joining Date
6/2/1997	14/6/2022

COURSE INFORMATION

Department	Section
IT	1

CONTACT INFORMATION

Figure 46: UI for student profile

SMS

- Overview
- Profile
- Attendance
- TimeTable
- Marks
- Fee History
- Upload Assignments**
- Assignments
- Settings
- Logout

Upload Assignment and Home Works

Upload your Assignment As a PDF

Choose File No file chosen

Submit

myFile-UX-Enhancement.pdf.pdf

1 / 2 | - 90% + ⌂ ⌃ ⌄

UX Designer Assessment

Problem Statement
Produce an IT Incident(ticket) Management System such as IT helpdesk application for an organization of your choice.

Imagine there is a manual process in place now. Assume there are multiple type of

Figure 47: UI for student upload assignments and homework

The screenshot shows the SMS student view interface. On the left is a dark sidebar with white icons and text:

- Overview
- Profile
- Attendance
- TimeTable
- Marks
- Fee History
- Upload Assignments
- Assignments** (highlighted in yellow)
- Settings
- Logout

The main area has a black header with three horizontal bars. Below it is a light gray box containing:

Assignments you have to done

ASSIGNMENT INFORMATION

#	Assignment Name	View
0	myFile-ITM 12_The Elite_Final report (2).pdf.pdf	View

Figure 48: UI for student view assignments and homework

The screenshot shows the "View Assignment" page. At the top, it says "View Assignment" in red. Below that is the file name "myFile-ITM 12_The Elite_Final report (2).pdf.pdf". The page includes a zoom control (1 / 98 | - 90% +) and a toolbar with download, print, and other icons.

The main content area displays the PDF document. The first page is titled "Final Report" and "Level 4". It also mentions "Discussion Forum Grading Tool for Learning Management System" and "The Elite". The second page is titled "Group Members:".

Figure 49: UI for student view and download assignments and homework

5.1.3. Parent Module

The screenshot shows a 'parent LOGIN' page with a dark header bar containing the text 'SMS'. Below the header is a light gray background area. In the center, there is a white rectangular form titled 'Login' with a small icon of a person and a key. The form contains two input fields: 'Email' with placeholder text 'Enter Email' and 'Password' with placeholder text 'Enter Password'. Below these fields is a blue rectangular button labeled 'Login'. Underneath the button, there is a link 'Forgot Password? [Generate New Password](#)'.

Figure 50: UI for parent login

The screenshot shows a 'Forgot your password?' page with a dark header bar containing the text 'SMS'. Below the header is a light gray background area. In the center, there is a white rectangular form. At the top, it says 'Forgot your password?'. Below that, a message reads 'Change your password in three easy steps. This will help you to secure your password!'. Three numbered steps are listed: 1. Enter your email address below., 2. Our system will send you a temporary link, 3. Use the link to reset your password. There is an input field labeled 'Enter your email address' with a placeholder 'Enter the email address you used during the registration. Then we'll email a link to this address.' Below the input field is a blue rectangular button labeled 'Get New Password'. At the bottom of the form is a red rectangular button labeled 'Back to Login'.

Figure 51: UI for parent forgot password

SMS

Forgot your password?

Change your password in three easy steps. This will help you to secure your password!

1. Enter your email address below.
2. Our system will send you a temporary link.
3. Use the link to reset your password.

Reset Link Sent Successfully! ×

Enter your email address

Enter the email address you used during the registration. Then we'll email a link to this address.

Get New Password

[Back to Login](#)

Figure 52: UI for parent password resetting link

SMS

☰

Hello Janitha Welagedara



**My
Attendance**

Attendance

View the attendance status for each of your courses. (Make sure that the atted. is atleast 75%)

View Attendance



Marks

View the marks obtained for each of your courses. These include the marks upto your previous semester.

View Marks



Time Table

View your timetable. (It shows the time table of all the courses you have registered for!)

View Timetable

Figure 53: UI for parent overview

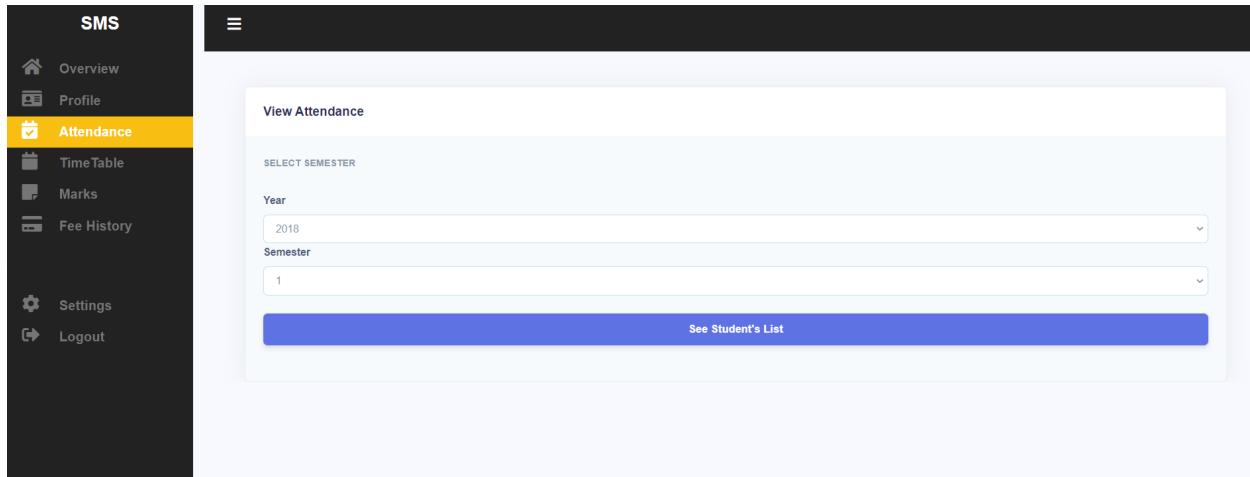


Figure 54: UI for view student's attendance

5.1.4. Teacher Module

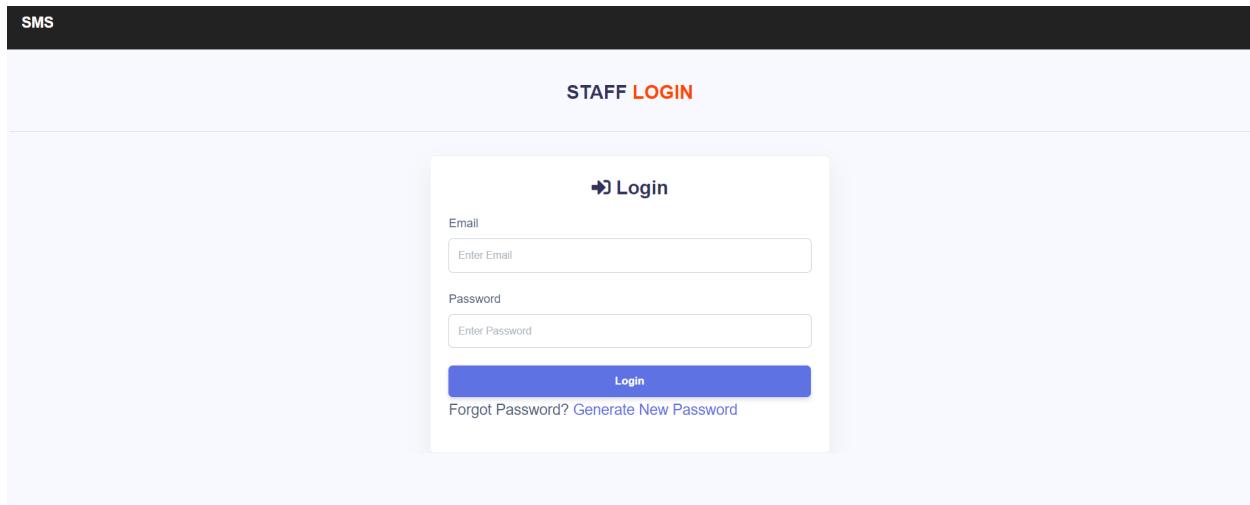
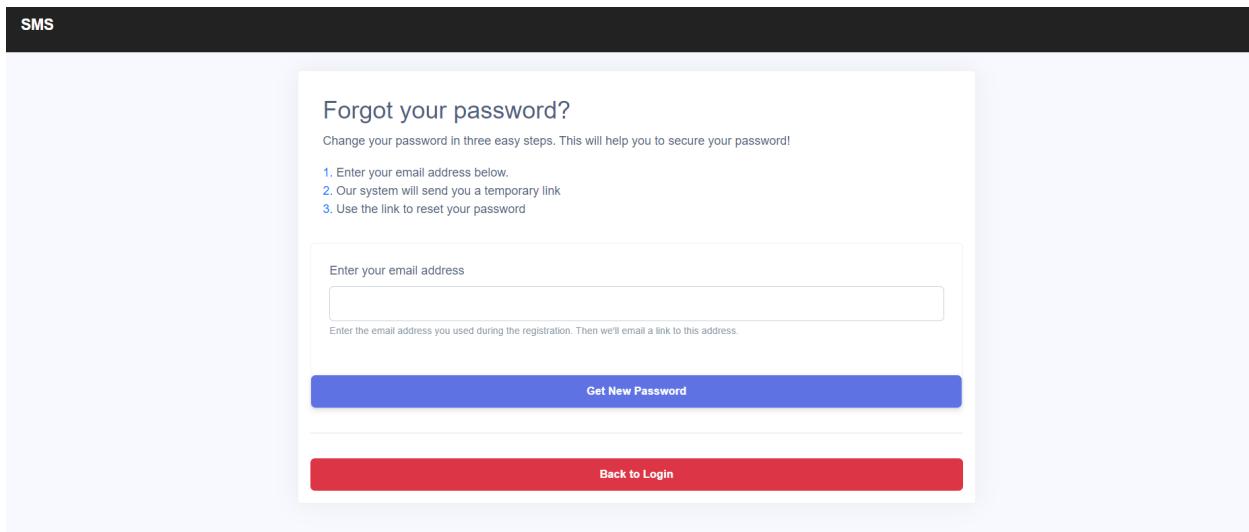
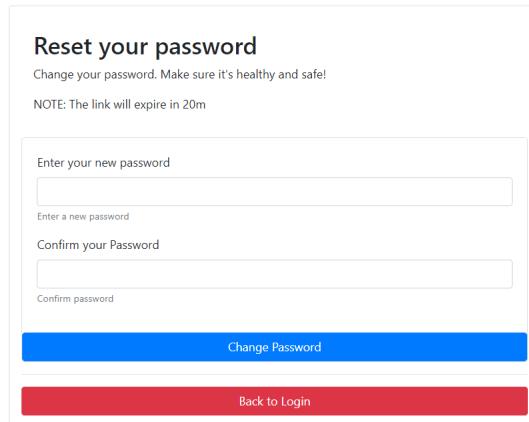


Figure 55: UI for teacher login



The screenshot shows a 'Forgot your password?' page. At the top, it says 'Forgot your password?'. Below that, a note reads: 'Change your password in three easy steps. This will help you to secure your password!' followed by a numbered list: 1. Enter your email address below., 2. Our system will send you a temporary link, 3. Use the link to reset your password. There is an input field labeled 'Enter your email address' with a placeholder 'Enter the email address you used during the registration. Then we'll email a link to this address.' Below the input field is a blue button labeled 'Get New Password'. At the bottom is a red 'Back to Login' button.

Figure 56: UI for teacher forgot password



The screenshot shows a 'Reset your password' page. At the top, it says 'Reset your password'. Below that, a note reads: 'Change your password. Make sure it's healthy and safe!' followed by a note: 'NOTE: The link will expire in 20m'. There are two input fields: 'Enter your new password' and 'Confirm your Password', both with placeholder text 'Enter a new password' and 'Confirm password' respectively. Below the inputs is a blue button labeled 'Change Password'. At the bottom is a red 'Back to Login' button.

Figure 57: UI for teacher reset password

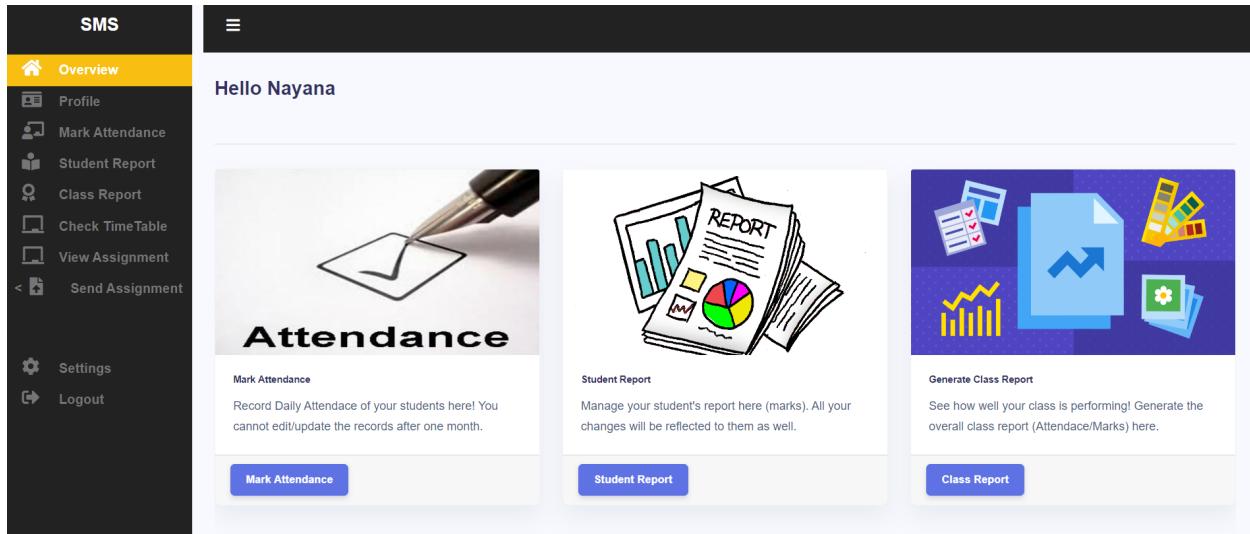


Figure 58: UI for teacher overview

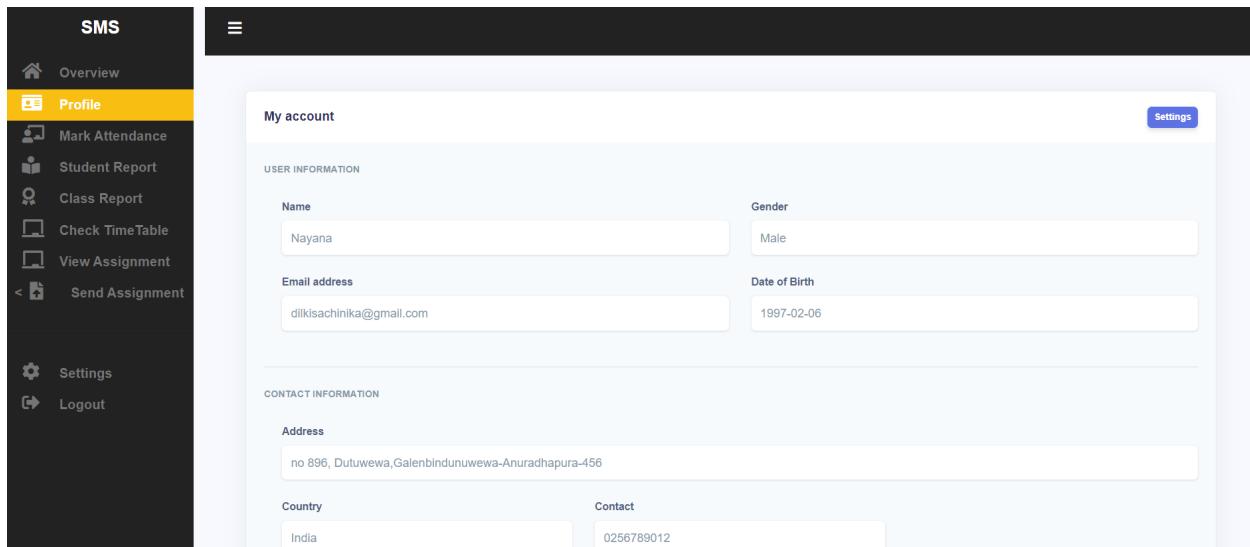


Figure 59: UI for teacher profile

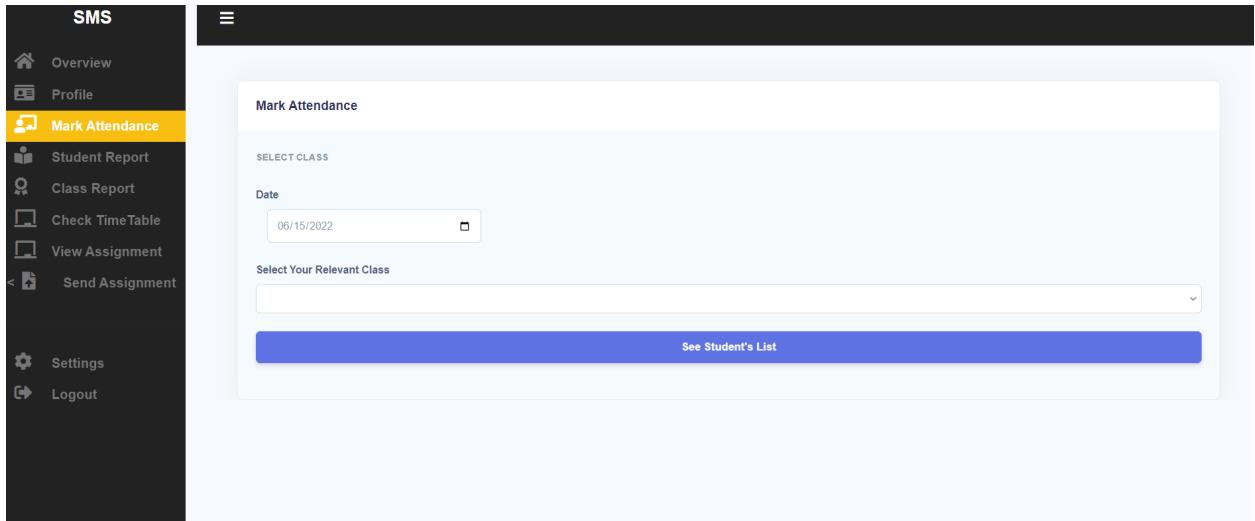


Figure 60: UI for teacher mark attendance

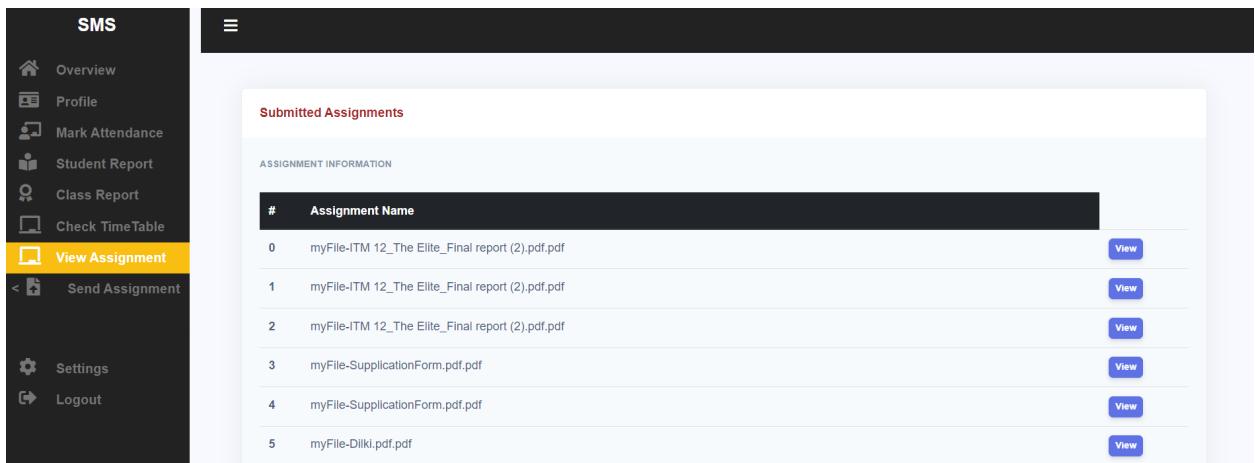


Figure 61: UI for teacher view submitted assignments

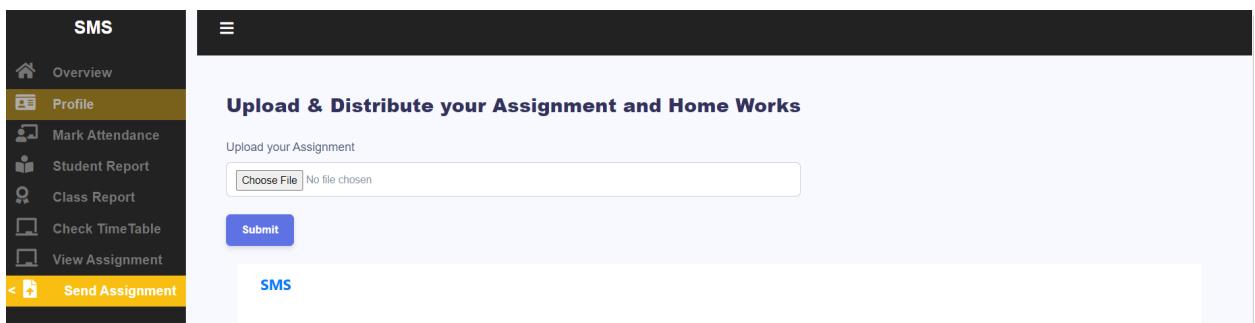


Figure 62: UI for teacher upload assignments and homework

5.1.2. Code segments

```
1  use smsdbms;
2
3  CREATE TABLE IF NOT EXISTS `admin`(
4      `admin_id` VARCHAR(36) NOT NULL,
5      `name` VARCHAR(255) NOT NULL,
6      `email` VARCHAR(255) NOT NULL UNIQUE,
7      `password` VARCHAR(255) NOT NULL,
8      PRIMARY KEY(`admin_id`)
9  );
10
11 CREATE TABLE IF NOT EXISTS `course` (
12     `c_id` VARCHAR(100) NOT NULL UNIQUE,
13     `semester` INT NOT NULL,
14     `name` VARCHAR(255) NOT NULL,
15     `c_type` VARCHAR(255) NOT NULL,
16     `credits` INT NOT NULL,
17     `dept_id` VARCHAR(255) NOT NULL,
18     PRIMARY KEY (`c_id`)
19 );
20
21 CREATE TABLE IF NOT EXISTS `student` (
22     `s_id` VARCHAR(36) NOT NULL,
23     `s_name` VARCHAR(255) NOT NULL,
24     `gender` VARCHAR(6) NOT NULL,
25     `dob` DATE NOT NULL,
26     `email` VARCHAR(255) NOT NULL UNIQUE,
27     `s_address` VARCHAR(255) NOT NULL,
28     `contact` VARCHAR(12) NOT NULL,
29     `password` VARCHAR(255) NOT NULL,
30     `section` INT NOT NULL,
```

Figure 63: Code segment for the database

```
1  const mysql = require('mysql');
2
3  module.exports = class Mysql {
4      static connect() {
5          // establish connection
6          const db = mysql.createConnection({
7              host: process.env.DB_HOST,
8              user: process.env.DB_USER,
9              password: process.env.DB_PASS,
10             database: 'smsdbms',
11         });
12         // connect to database
13         db.connect((err) => {
14             if (err) {
15                 throw err;
16             }
17             console.log('Mysql Connected');
18         });
19         return db;
20     }
21 };
22
```

Figure 64: Code segment for the database

```
controllers > JS admin.js > ...
1  const mysql = require('mysql');
2  const jwt = require('jsonwebtoken');
3  const bcrypt = require('bcryptjs');
4  const uuidv4 = require('uuid').v4;
5  const mailgun = require('mailgun-js');
6  let ejs = require("ejs");
7  let pdf = require("html-pdf");
8  let path = require("path");
9  const fs = require("fs");
10 const DOMAIN = process.env.DOMAIN_NAME;
11 const mg = mailgun({apiKey: process.env.MAILGUN_API_KEY, domain: DOMAIN});
12
13 const db = mysql.createConnection({
14   host: process.env.DB_HOST,
15   user: process.env.DB_USER,
16   password: process.env.DB_PASS,
17   dateStrings: 'date',
18   database: 'smsdbms',
19 });
20
21 // Students limit per section
22 const SECTION_LIMIT = 20;
23
24 // Database query promises
25 const zeroParamPromise = (sql) => {
26   return new Promise((resolve, reject) => {
27     db.query(sql, (err, results) => {
28       if (err) return reject(err);
29       return resolve(results);
30     });
31   });
32 }
```

Figure 65: Code segment for the database

```
controllers > JS parent.js > ...
1  const mysql = require('mysql');
2  const jwt = require('jsonwebtoken');
3  const bcrypt = require('bcryptjs');
4
5  const mailgun = require('mailgun-js');
6  const DOMAIN = process.env.DOMAIN_NAME;
7  const mg = mailgun({ apiKey: process.env.MAILGUN_API_KEY, domain: DOMAIN });
8
9  const db = mysql.createConnection({
10    host: process.env.DB_HOST,
11    user: process.env.DB_USER,
12    password: process.env.DB_PASS,
13    dateStrings: 'date',
14    database: 'smsdbms',
15  });
16
17 // Database query promises
18 const zeroParamPromise = (sql) => {
19   return new Promise((resolve, reject) => {
20     db.query(sql, (err, results) => {
21       if (err) return reject(err);
22       return resolve(results);
23     });
24   });
25 };
26
27 const queryParamPromise = (sql, queryParam) => {
28   return new Promise((resolve, reject) => {
29     db.query(sql, queryParam, (err, results) => {
30       if (err) return reject(err);
31     });
32   });
33 };
34
35 module.exports = { zeroParamPromise, queryParamPromise };
```

Figure 66: Code segment controllers - Parent

```

controllers > JS staff.js > ...
1  const mysql = require('mysql');
2  const jwt = require('jsonwebtoken');
3  const bcrypt = require('bcryptjs');
4
5  const fs= require('fs');
6  const path=require('path');
7  // const dirPath= path.join(__dirname + '/public','doc');
8  // console.log("dirPath",dirPath);
9
10 const cur_dir = process.cwd();
11 const dirPath= path.join(cur_dir + '/public','doc');
12 let uploadFile;
13 // change current directory
14
15 // process.chdir('E:\\My_works\\17_batch\\new\\sms-backend');
16 // console.log("process.cwd() ",process.cwd());
17 //
18 // // path.join for cross-platform paths:
19 // process.chdir(path.join('E:', 'My_works', '17_batch'));
20 // console.log("process.cwd() 2 ",process.cwd());
21
22 const mailgun = require('mailgun-js');
23 const multer = require("multer");
24 const {v4: uuidv4} = require("uuid");
25 const DOMAIN = process.env.DOMAIN_NAME;
26 const mg = mailgun({ apiKey: process.env.MAILGUN_API_KEY, domain: DOMAIN });
27
28
29
30 const db = mysql.createConnection({

```

Figure 67: Code segment controllers - Staff

```
controllers > js student.js > ...
1  const mysql = require('mysql');
2  const jwt = require('jsonwebtoken');
3  const bcrypt = require('bcryptjs');
4
5  const multer = require('multer');
6  const mailgun = require('mailgun-js');
7
8  const fs = require('fs');
9  const path = require('path');
10 const {v4: uuidv4} = require("uuid");
11
12 const DOMAIN = process.env.DOMAIN_NAME;
13 const mg = mailgun({ apiKey: process.env.MAILGUN_API_KEY, domain: DOMAIN });
14
15 let uploadFile;
16
17
18 const db = mysql.createConnection({
19   host: process.env.DB_HOST,
20   user: process.env.DB_USER,
21   password: process.env.DB_PASS,
22   dateStrings: 'date',
23   database: 'smsdbms',
24 });
25
26 // Database query promises
27 const zeroParamPromise = (sql) => {
28   return new Promise((resolve, reject) => {
29     db.query(sql, (err, results) => {
30       if (err) return reject(err);
```

Figure 68: Code segment controllers - Student

```

middlewares > JS adminAuth.js > ...
1  const jwt = require('jsonwebtoken');
2  const mysql = require('mysql');
3
4  const db = mysql.createConnection({
5    host: process.env.DB_HOST,
6    user: process.env.DB_USER,
7    password: process.env.DB_PASS,
8    dateStrings: 'date',
9    database: 'smsdbms',
10   });
11
12 const selectID = (id) => {
13   return new Promise((resolve, reject) => {
14     const sql1 = 'SELECT name FROM admin WHERE admin_id = ?';
15     db.query(sql1, [id], (err, results) => {
16       if (err) return reject(err);
17       return resolve(results);
18     });
19   });
20 };
21
22 const requireAuth = (req, res, next) => {
23   const token = req.cookies.jwt;
24   if (token) {
25     jwt.verify(token, process.env.JWT_SECRET, async (err, result) => {
26       if (err) {
27         req.flash(
28           'error_msg',
29           'You need to login as ADMIN in order to view that source!'
30         );
31     });
32   }
33 };
34
35 module.exports = { selectID, requireAuth };

```

Figure 69: Code segment Authentication - Admin

```

middlewares > JS parentAuth.js > ...
1  const jwt = require('jsonwebtoken');
2  const mysql = require('mysql');
3
4  const db = mysql.createConnection({
5    host: process.env.DB_HOST,
6    user: process.env.DB_USER,
7    password: process.env.DB_PASS,
8    dateStrings: 'date',
9    database: 'smsdbms',
10   });
11
12 const selectID = (id) => {
13   return new Promise((resolve, reject) => {
14     const sql1 = 'SELECT p_name FROM parent WHERE p_id = ?';
15     db.query(sql1, [id], (err, results) => {
16       if (err) return reject(err);
17       return resolve(results);
18     });
19   });
20 };
21
22 const requireAuth = (req, res, next) => {
23   const token = req.cookies.jwt;
24   if (token) {
25     jwt.verify(token, process.env.JWT_SECRET, async (err, result) => {
26       if (err) {
27         req.flash(
28           'error_msg',
29           'You need to login as PARENT in order to view that source! ==> '
30         );
31     });
32   }
33 };
34
35 module.exports = { selectID, requireAuth };

```

Figure 70: Code segment Authentication - Parent

```
middlewares > JS staffAuth.js > ...
1  const jwt = require('jsonwebtoken');
2  const mysql = require('mysql');
3
4  const db = mysql.createConnection({
5    host: process.env.DB_HOST,
6    user: process.env.DB_USER,
7    password: process.env.DB_PASS,
8    dateStrings: 'date',
9    database: 'smsdbms',
10   });
11
12  const selectID = (id) => {
13    return new Promise((resolve, reject) => {
14      const sql1 = 'SELECT st_name FROM staff WHERE st_id = ?';
15      db.query(sql1, [id], (err, results) => {
16        if (err) return reject(err);
17        return resolve(results);
18      });
19    });
20  };
21
22  const requireAuth = (req, res, next) => {
23    const token = req.cookies.jwt;
24    if (token) {
25      jwt.verify(token, process.env.JWT_SECRET, async (err, result) => {
26        if (err) {
27          req.flash(
28            'error_msg',
29            'You need to login as STAFF in order to view that source!'
30          );
31        }
32      });
33    }
34  };
35
36  module.exports = { selectID, requireAuth };
```

Figure 71: Code segment Authentication - Staff

```

middlewares > JS studentAuth.js > ...
1  const jwt = require('jsonwebtoken');
2  const mysql = require('mysql');
3
4  const db = mysql.createConnection({
5    host: process.env.DB_HOST,
6    user: process.env.DB_USER,
7    password: process.env.DB_PASS,
8    dateStrings: 'date',
9    database: 'smsdbms',
10   });
11
12  const selectID = (id) => {
13    return new Promise((resolve, reject) => {
14      const sql1 = 'SELECT s_name FROM student WHERE s_id = ?';
15      db.query(sql1, [id], (err, results) => {
16        if (err) return reject(err);
17        return resolve(results);
18      });
19    });
20  };
21
22  const requireAuth = (req, res, next) => {
23    const token = req.cookies.jwt;
24    if (token) {
25      jwt.verify(token, process.env.JWT_SECRET, async (err, result) => {
26        if (err) {
27          req.flash(
28            'error_msg',
29            'You need to login as STUDENT in order to view that source! ==> '
30          );

```

Figure 72: Code segment Authentication - Student

6.2 Summary

In this chapter we explained the implementations that we have done up to now. Currently we are developing the database and designing web and interfaces.

Chapter 6

Discussion and Conclusion

In here we hope to provide a software solution for a school to manage their day today activities which are done at the school. In system there are four roles. They are Admin, Teacher, Parent and Student. They have different kind of functionalities. As technologies we have used EJS, JavaScript, HTML and CSS. and Node JS. For the database we have used MySql. Through the proposed system the drawbacks in the current school management systems will be avoided while establishing a strong relationship among students, parents, teachers and school administration.

References

- 1.<https://www.javacodegeeks.com/2016/05/limitations-monolithic-application-need-adaptingmicroservices-architecture.html>
- 2.<https://dzone.com/articles/why-microservices>
- 3.<https://articles.microservices.com/>
- 4.<https://www.aspicio.com/>
- 5.<https://spring.io/projects/spring-boot>

Appendix A

Individual Contribution to the Project

175083J-Thilakarathna W.M.D.S.

I'm responsible to develop the student role in "School Management System". The student can log in to the system or register with the system. When he is login to the system, he can put the user's name and the password. Then the student can edit the profile also. When it comes to assignments and homework, when teacher is uploading assignments and homework, the student can view. As well as student can upload answers both assignments and home works. When teacher or admin add some kind of notices student can view those notices. Student is doing creative activities. Then student can upload those creative activities via the system. When teacher is uploading some quiz, the student can access to the quiz and do the quiz with the system and he can submit the quizzes via the system. As well as student can View Assignment marks, timetable, progress reports, attendance report and upcoming events in the school. I'm contributing to both backend and frontend for the "Student module". Rather than that I contributed to draw diagrams also.

175090D-Welagedara J.M.

I'm responsible to develop the parent role in "School Management System". They can log in to the system or register with the system. When parent is login to the system, he can put the user's name and the password. Then the student can edit the profile also. Parent can view the time table, view the marks of students, view the dashboard. As well as the parent can view the creative activities of the student. When the admin adds the progress report of the student, the parent can view the progress report also. The parent can view the attendance of the particular student. When the school is requesting some donations from parents, parent can do donations. As well as the parent can pay school fees. The parent can chat with the teacher. I'm contributing to both backend and frontend for the "Teacher module." Rather than that I contributed to draw diagrams also.

175064D-Prasandika K.A.C

As the member of the school management system project, first of all, we discussed with the client, and we got a software project. Before starting implementation, we divided our project into 4 modules. Then I assigned the teacher module. I draw activity diagrams and class diagrams, and the functionalities associated with my module are: marking attendance, uploading quizzes, uploading notices, uploading marks, etc. I learned about React and NodeJS and am continuing to work.

175053T-Nasrina M.I.F.

I am responsible for admin module in the system. I am contributing to design front end and backend part of admin module.

We together designed all the design diagrams, database and all the documentation works. The mainly assigned diagram for me was Class diagram and I developed it well and I developed activity diagrams to my module in our system. The functionalities assigned for me is Admin module such as: add profiles, upload time table and course information (subjects), Generating various type of report as attendance report, progress report, uploading notices view notices and handling payments. For now, I designed user interface of admin and I have finished the front end works for that module. In backend I have working for the admin module. Now I am continuing with the further works assigned for me.